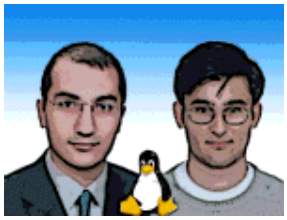
LinuxFocus article number 268
http://linuxfocus.org



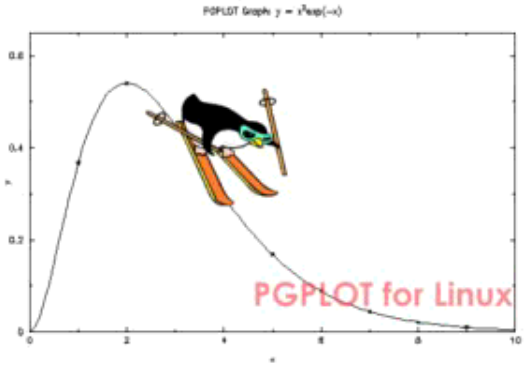
# Utiliser PGPLOT pour du graphique interactif sous Linux

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

*L´auteur:*

Nous sommes deux assistants de recherche à l'Institut d'Informatique de l'université?d'ITU. Nous réalisons des applications d'ingénierie et nous utilisons Linux pour cela... La page personnelle de Seckin se trouve à^ www.be.itu.edu.tr/~seckin

*Traduit en Français par:*
Paul Delannoy (homepage)




*Résumé*:

Cet article va vous apprendre comment ?écrire un programme graphique interactif en Fortran. PGPlot est l'ensemble des sous-programmes nécessaires à l'utilisation du code Fortran. Nous allons décrire leur installation et l'écriture de quelques exemples. Deux de ces exemples seront analysés, ainsi que leur code Fortran, ce qui vous donnera une idée des applications que vous pourrez réaliser avec PgPlot.

# Introduction

## Qu'est-ce que PGPLOT?

La bibliothèque de programmes graphiques PGPLOT est un outil logiciel graphique utilisable en Fortran ou en C, indépendant des périphériques, qui permet de réaliser des graphes scientifiques simples. Il est conçu pour permettre la création de graphiques de qualité et publiables avec un minimum d'efforts. Dans la plupart des cas le programme peut être indépendant du périphérique et ses sorties sont dirigées vers le périphérique approprié lors de son exécution.

La bibliothèque PGPLOT se compose de deux grandes parties : l'une concerne ce qui est indépendant du

périphérique, l'autre concerne ce qui est dépendant du périphérique et propose un ensemble de pilotes, sous-programmes effectuant les sorties sur différents terminaux, ?écrans d'affichage, imprimantes à aiguilles ou à jet d'encre, imprimantes laser, et traceurs de courbes. Les format de fichiers les plus répandus sont supportés, y compris PostScript et GIF. PGPLOT elle-même est écrite en Fortran-77 standard.

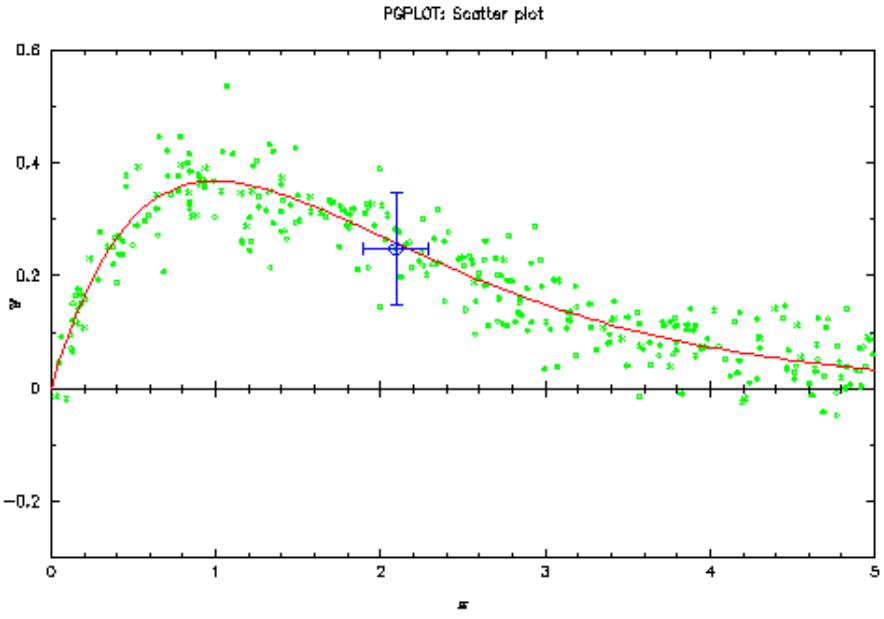
Les routines PGPLOT peuvent ?être invoquées directement dans un programme Fortran-77 ou Fortran-90. Une bibliothèque d'interface en C (cpgplot) et un fichier d'en-tête (cpgplot.h) sont fournis afin d'assurer les transferts lors d'appels depuis un programme en C ou C++; cette bibliothèque effectue les conversions nécessaires entre C et Fortran. PGPLOT a été?testé sous UNIX (plusieurs variétés, dont Linux, SunOS, Solaris, HPUX, AIX, et Irix) et OpenVMS.

PGPLOT n'est pas dans le domaine public. Pourtant il est disponible librement pour des usages non-commerciaux. Le code source et la documentation, ainsi qu'un petit nombre de routines spécifiques à^ certains systèmes, sont la propriété du California Institute of Technology (CalTech). Pour disposer du fichier et des instructions d'installation cliquez ici.

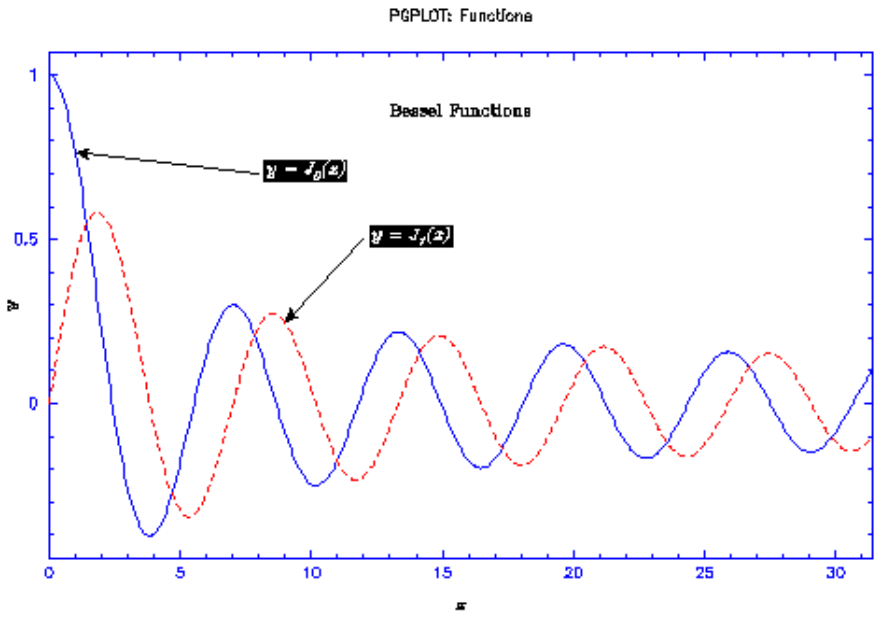
## Quelques exemples

Ci-dessous quelques exemples d'application de PGPLOT afin de montrer ses capacités.

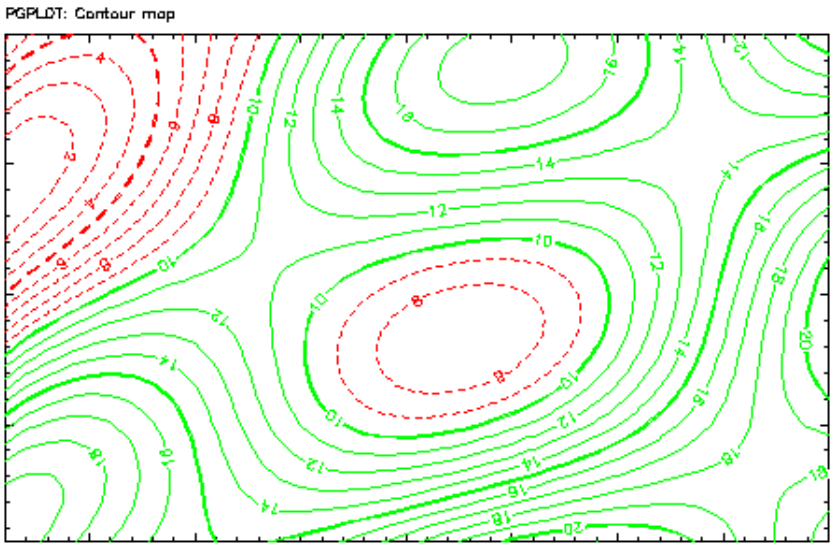
Exemple 1) Nuage de points :



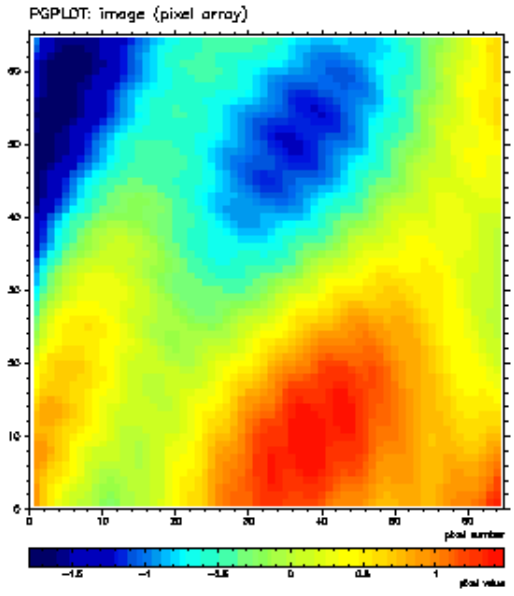

Exemple 2) Graphe de fonction :



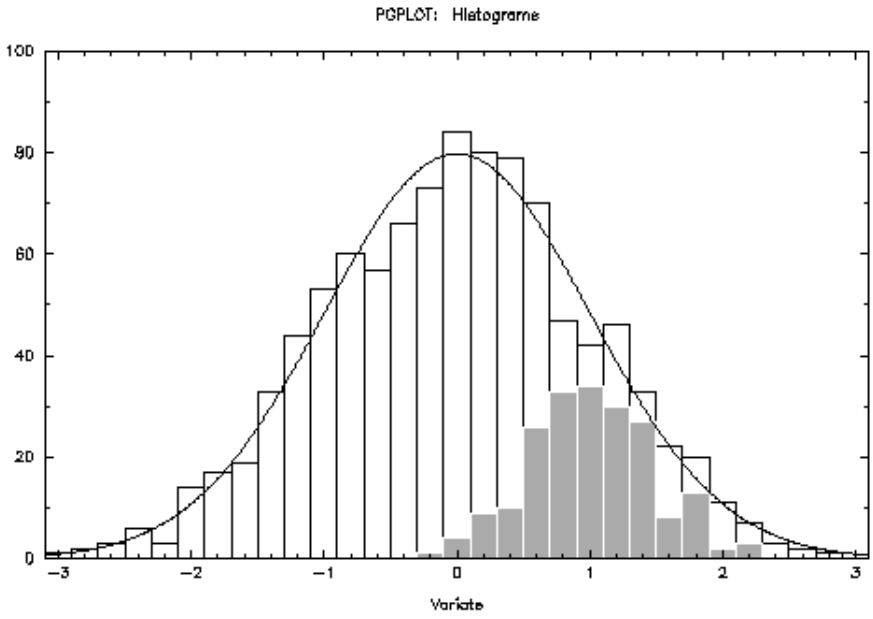

Exemple 3) Contours :




Exemple 4) Images :




Exemple 5) Histogrammes:




# Installation pour un système Unix

**Note :** On parle ici de deux répertoires, le *répertoire de distribution (source)* qui devra contenir l'arborescence des sources de PGPLOT et le *répertoire cible* dans lequel vont être créés les bibliothèques propres aux machines, les fichiers de données, et les programmes de démonstration. Il est recommandé de créer des répertoires, *vides* dans ce but. Ils ne doivent pas?être identiques. Ci-dessous dans les exemples, ces répertoires sont nommés

`/usr/local/src/pgplot` (répertoire de distribution)
`/usr/local/pgplot` (répertoire cible)

mais vous pouvez choisir d'autres noms. Ce n'est pas courant mais les privilèges root ne sont pas nécessaires pour installer PGPLOT, il suffit d'avoir le droit d'écriture dans ces répertoires. Le même répertoire de distribution peut servir pour l'installation de plusieurs binaires de PGPLOT destinés à des architectures distinctes, mais dans des répertoires cibles différents.

L'installation sous Linux est ici présentée brièvement, si vous avez des questions complémentaires vous pouvez envoyer un courrier électronique à tjp(At)astro.caltech.edu

## Copier le fichier de distribution

Copiez le fichier de distribution par ftp anonyme depuis Caltech. Faites un ftp (user: anonymous, password: votre@adresse de courrier électronique) sur ftp.astro.caltech.edu. Le fichier de distribution est une archive UNIX tar compressée par gzip. Voici les commandes ftp nécessaires à la récupération du fichier :

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

Les fichiers texte évoqués ici sont également inclus dans l'archive `tar`.

Ce fichier peut?également être obtenu depuis l'URL
`ftp://ftp.astro.caltech.edu/pub/pgplot/pgplot5.2.tar.gz`.

## Décompresser les fichiers

Utilisez `gunzip` et `tar` pour décompresser l'archive et extraire son contenu. Cela va créer le répertoire pgplot (et ses sous-répertoires) dans le répertoire courant. Assurez-vous donc que le répertoire courant est bien celui où vous voulez créer l'arborescence de distribution de PGPLOT.

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

Cet exemple crée `/usr/local/src/pgplot` et ses sous-répertoires.

## Créer le répertoire cible

Créez un répertoire avec droit d'écriture, dans lequel seront placés la bibliothèque PGPLOT et ses fichiers associés. Un tel répertoire est nécessaire pour chaque combinaison système d'exploitation-compilateur ("système cible") pour lequel vous souhaitez installer PGPLOT, par exemple :

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

N’essayez pas de créer la bibliothèque PGPLOT dans le répertoire de distribution.

## Sélection des pilotes de périphériques

Vous allez configurer PGPLOT en choisissant les périphériques dans la liste proposée. Copiez le fichier `drivers.list` du répertoire de distribution vers le répertoire cible, puis utilisez un éditeur de texte pour choisir les pilotes. Chaque pilote de périphérique disponible figure dans une ligne du fichier : l’abscence de point d’exclamation (!) au début de la ligne active le pilote, par contre sa présence le désactive. Plusieurs d’entre eux sont spécifiques à un système d’exploitation (voir les notes dans `drivers.list`), aussi n’activez **que les périphériques que vous utiliserez vraiment**. PGPLOT pourra être reconfiguré plus tard en reprenant l’installation à cette étape. Vous devrez souvent inclure : le périphérique null (/NULL), des imprimantes PostScript (/PS, /VPS, /CPS, et /VCPS), des terminaux Tektronix (/TEK, /XTERM, et d’autres variantes), et, si le système cible dispose du X Window System, les pilotes X Window (/XWINDOW, /XSERVE). Vous pouvez aussi inclure les pilotes pour les fichiers GIF (/GIF, /VGIF) ou d’autres imprimantes.

```
cd /usr/local/pgplot
cp /usr/local/src/pgplot/drivers.list .
vi drivers.list  (ou votre éditeur favori)
```

## Créer le fichier makefile

L’installation sous UNIX de PGPLOT utilise un script, nommé `makemake`, pour générer un fichier makefile UNIX standard pour votre système, vos compilateurs, et les périphériques PGPLOT choisis. L’information sur le système et le compilateur est fournie par un *fichier de configuration*. Des fichiers de configuration sont disponibles pour les systèmes qui suivent. Si le votre n’est pas dans la liste, ou si vous rencontrez des problèmes pour créer le makefile, vous trouverez ci-dessous la façon de créer votre propre fichier de configuration.

Notez que chaque fichier de configuration est établi pour UN compilateur donné fonctionnant sur UN système d’exploitation donné. Si, par exemple, sur votre système, la commande `f77` appelle le compilateur GNU g77, vous ne pouvez pas utiliser le fichier de configuration d’un compilateur SPARC f77, par exemple. Vous aurez à créer un fichier de configuration particulier. Dans le tableau suivant, Arg#2 est le code correspondant au système d’exploitation, et Arg#3 est le code correspondant aux compilateurs Fortran et C. Vous obtiendrez des informations supplémentaires sur les systèmes supportés en lisant le fichier `pgplot/sys_*/aaaread.me`, où * désigne une des valeurs possibles de Arg#2.

```
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
```

Si votre système est dans la liste, faites comme suit : faites du répertoire cible votre répertoire courant par défaut, comme ceci,

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

Exécutez le script `makemake` depuis le répertoire de distribution :

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

Le premier argument donné à `makemake` est le nom du répertoire de distribution. Notez que lorsque vous lancez `makemake`, le répertoire courant par défaut doit être le répertoire cible, c-à-d. le répertoire dans lequel vous voulez mettre les bibliothèques compilées.

Le second argument est le nom du système d'exploitation (Arg#2 dans la table précédente); si vous l'omettez ou si vous choisissez un nom erroné, makemake vous donnera la liste des valeurs admises. Sur certains systèmes, qui proposent plusieurs compilateurs Fortran ou C, un troisième argument est nécessaire (Arg#3 dans la table précédente); il est composé des noms des deux compilateurs réunis par un caractère de soulignement. Si vous l'omettez, makemake vous donnera la liste des valeurs admises.

Même si vos arguments sont les bons, makemake peut encore se plaindre de ne pas trouver la liste des pilotes actifs (`drivers.list`). Reprenez à l'étape 4!

**Exemple**

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.

Le script `makemake` génère un fichier `makefile` qui sera utile plus tard, un fichier Fortran `grexec.f` chargé des appels aux pilotes sélectionnés, et un fichier texte `rgb.txt` qui contient les codes couleur nécessaires à la routine PGSCRN. (Si un tel fichier `rgb.txt` existait auparavant, avec vos propres définitions de couleurs, `makemake` ne le modifierait pas.) Il copie aussi deux fichiers en-tête en Fortran nécessaires à la compilation. Ainsi, à ce stade, vous devriez au moins avoir les fichiers :

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

Vérifiez qu'ils ont bien été créés et que la liste des pilotes que `makemake` déclare avoir trouvé est bien identique à celle indiquée dans `drivers.list`. Si votre UNIX n'est pas dans la liste des systèmes supportés ci-dessus, créez votre propre fichier de configuration dans le répertoire cible, en le nommant `local.conf`. Le mieux est de copier un des fichiers fournis (dans `pgplot/sys_*/*.conf`), et de l'éditer en suivant les commentaires présents dans ce fichier. La procédure `makemake` utilisera `local.conf` s'il existe dans le répertoire courant, et si vous ne spécifiez pas de valeur pour Arg#3. Mais Arg#2 est toujours nécessaire (OS).

## Taper 'make' pour compiler le code

La commande UNIX `make` va compiler le code de la bibliothèque PGPLOT en s'appuyant sur les données du fichier `makefile` :

make

Par défaut, `make` génère : une bibliothèque module-objet, `libpgplot.a`; une bibliothèque partagée (si le système le supporte), le fichier binaire de polices PGPLOT `grfont.dat`, les programmes de démonstration `pgdemo*`, et un fichier de documentation `pgplot.doc`. De plus, si les pilotes /XWINDOW et/ou /XSERVE ont été activés à l'étape 4, le programme `pgxwin_server` sera cr??, et si le pilote /XDISP a ?t? activ?, le programme `pgdisp` sera créé. Si cette étape se termine correctement, vous pouvez taper

make clean

pour effacer les fichiers intermédiaires devenus inutiles. Vous auriez alors au moins dans votre répertoire courant :

drivers.list
grexec.f
grfont.dat (fichier binaire de polices)*
libpgplot.a (bibliothèque PGPLOT)*
libpgplot.so (bibliothèque partagée, sous réserve)*
makefile
pgdemo1 ... pgdemo16 (programmes de démonstration)
pgdisp (nécessaire au pilote /XDISP)*

pgplot.doc (fichier ASCII de documentation)
pgxwin_server (nécessaire au pilote /XWINDOW)*
rgb.txt (base de données des noms de couleurs)*

Si vous avez activé XMDRIV ou TKDRIV, certains des fichiers suivants seront là aussi:

pgmdemo (programme de démo exécutable)
libXmPgplot.a (bibliothèque objet nécessaire aux applications PGPLOT/Motif)*
XmPgplot.h (fichier en-tête nécessaire aux applications PGPLOT/Motif)*
libtkpgplot.a (bibliothèque objet nécessaire aux applications PGPLOT/Tk)*
pgtkdemo (programme de démo exécutable)
pgtkdemo.tcl (script nécessaire au programme de démo)
tkpgplot.h (fichier en-tête nécessaire aux applications PGPLOT/Tk)*

Pour déplacer PGPLOT dans un autre répertoire, vous devez y copier au minimum les fichiers indiqués par un astérisque (*). Le fichier de documentation contient les descriptions des routines PGPLOT, elles sont aussi fournies dans le manuel.

## Exécuter les programmes de démonstration

Exécutez les programmes de démonstration sur vos périphériques afin de vérifier qu'ils fonctionnent de manière satisfaisante. Avant toute exécution d'un programme utilisant PGPLOT, vous devez vérifier que la variable `PGPLOT_DIR` est correctement définie. C'est le nom du répertoire dans lequel PGPLOT va chercher les fichiers `grfont.dat` et `rgb.txt` (à moins que les variables `PGPLOT_FONT` et `PGPLOT_RGB` n'aient reçu des valeurs destinées^à modifier ce comportement), et, si nécessaire, le programme X Window server `pgxwin_server` :

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

Il n'est pas nécessaire, mais pratique, de définir un périphérique par défaut pour PGPLOT avec la variable `PGPLOT_DEV`, par exemple :

UNIX csh or tcsh: setenv PGPLOT_DEV /xwindow

Si vous utilisez une bibliothèque partagée sous UNIX (par ex. sous Solaris 2.x), vous devrez sans doute aussi inclure le répertoire PGPLOT dans votre chemin de recherche, défini par la variable d'environnement `LD_LIBRARY_PATH`. Pour lancer un programme, tapez son nom (avec le répertoire si le répertoire courant n'est pas dans votre chemin de recherche) :

./pgdemo1

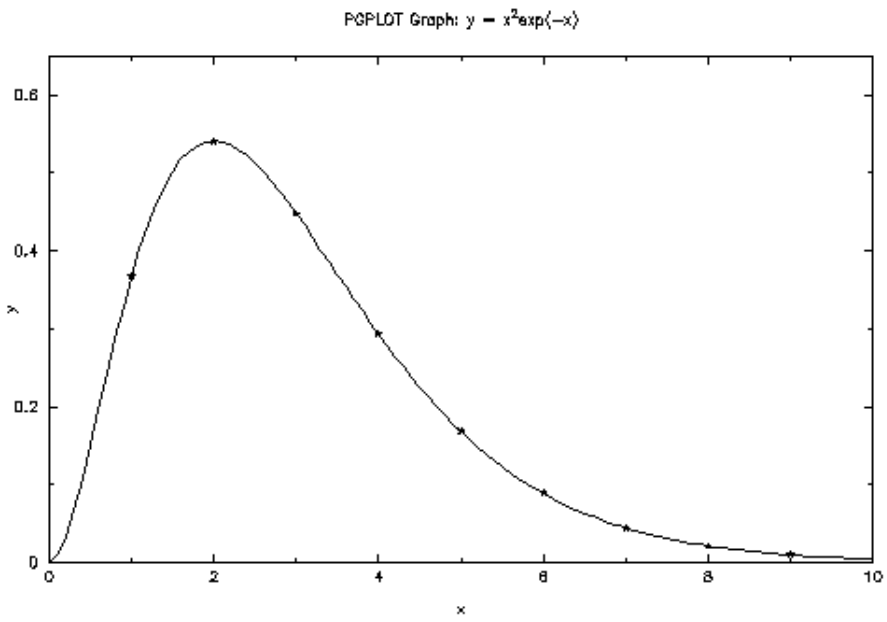
Tous ces programmes de démo demandent un périphérique et son type. Tapez un point d'interrogation `?` pour avoir la liste des pilotes disponibles et vérifier ainsi que PGPLOT est bien configuré. Points à vérifier : le programme PGPLOT lit correctement les polices et affiche correctement les exposants, les indices et les caractères spéciaux (pgdemo2) ; le programme PGPLOT lit correctement les définitions de couleurs (pgdemo10) ; sur les périphériques interactifs, le curseur est bien géré? (pgdemo5, pgdemo6).

# Comment compiler votre code ?

Après l'installation de la bibliothèque PGPLOT, vous pouvez utiliser ses routines dans votre propre code Fortran, comme ci-dessous :

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

C Calcule les coordonnées des points à tracer.

      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 Ouvre le périphérique graphique.

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

C Définit l'étendue du graphe (0 < x < 10, 0 < y < 0.65),
C et trace les axes.

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

C Etiquette les axes (notez l'usage de \u et \d pour les exposants).

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

C Trace le graphe.

      CALL PGLINE(101, XR, YR)

C Place des symboles aux points choisis.

      CALL PGPT(9, XS, YS, 18)

C Ferme le périphérique graphique.

      CALL PGCLOS

      END
```

Ceci affichera le graphique ci-dessous :




Pour que ce code s'exécute correctement, vous devrez lier la bibliothèque PGPLOT et les bibliothèques X11 dans votre code source. Le script suivant fait cette liaison :

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

Lors de l'exécution les fichiers suivants doivent être dans le répertoire courant :

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

Il suffit de les copier du répertoire pgplot vers le répertoire dans lequel vous exécutez votre code.

# Application 1: Cercle passant par 3 points

Notre but est de saisir les coordonnées de 3 points indiqués par des clics de souris, puis de laisser le code Fortran tracer le cercle qui passe par ces 3 points. Ce problème était une consigne donnée par le Dr.Serdar Celebi (mscelebi(at)itu.edu.tr) lors du cours de "Géométrie Computationnelle" que nous suivions.

Les routines suivantes définissent l'étendue et le fond de la zone dans laquelle vont figurer les tracés suivants. Leur usage est détaillé? dans le manuel PGPLOT. (voir le lien vers la page de PGPLOT à la fin)

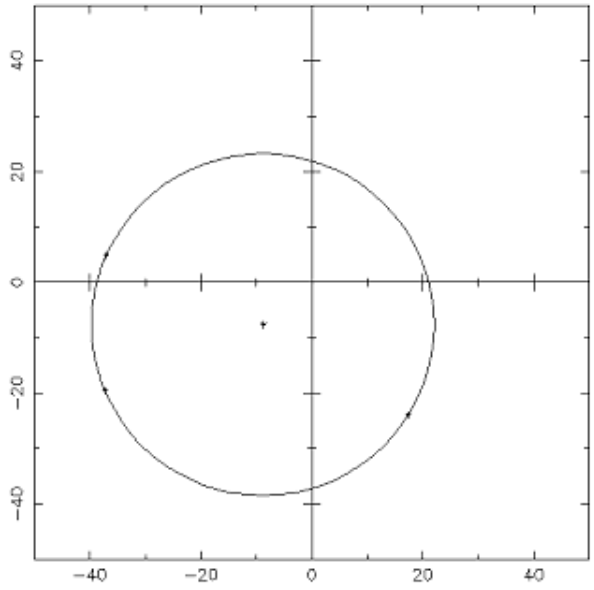
```
..
...
call PGSCR(0,1.0,1.0,1.0) !fixe les codes couleur
call PGENV(-50.0,50.0,-50.0,50.0,1,1) !définit fenêtre et étendue, dessine le cadre et son étiquette
call PGSCI(1) !fixe la couleur
call PGSFS(2) !fixe le style 'remplissage'
```

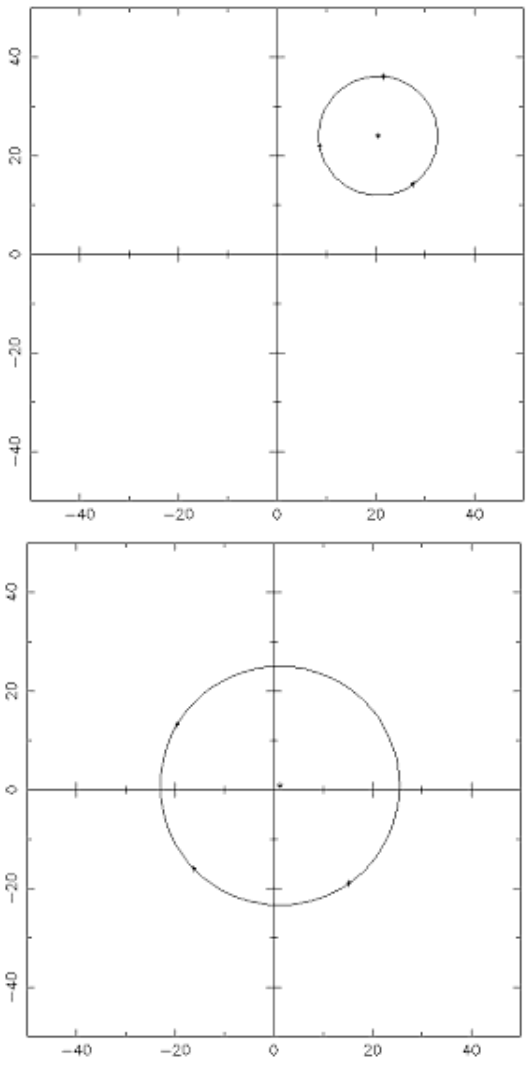
...
..

La routine utilisée, PGPT1, place un pointeur au point désigné par la souris.

```
..
...
WRITE (*,*) 'Mode curseur :', MODE
              GOTO 10
           END IF
            CALL PGPT1(X, Y, 3)  !trace un marqueur
         ic=ic+1
         xp(i)=x
         yp(i)=y
```

Après avoir marqué 3 points dans l'écran, le code Fortran calcule le centre puis trace le cercle qui passe par les 3 points.






```
..
...
c-----trouve le rayon---------------
 r=(xcenter-xp(1))**2+(ycenter-yp(1))**2
 r=r**0.5
c-----trace le cercle-------------------
 call PGCIRC(xcenter,ycenter,r) !trace un cercle
 goto 1
...
..
```

Nous traçons le cercle avec cette routine, "PGCIRC". Vous pouvez alors effacer le cercle pré-dessiné puis donner trois autres points et effectuer le tracé dans la même zone. Le code complet figure dans les références (cf. Réf. circle.f).

# Application 2: Tracer? des "épines de porc-épic" sur une courbe

# d’approximation

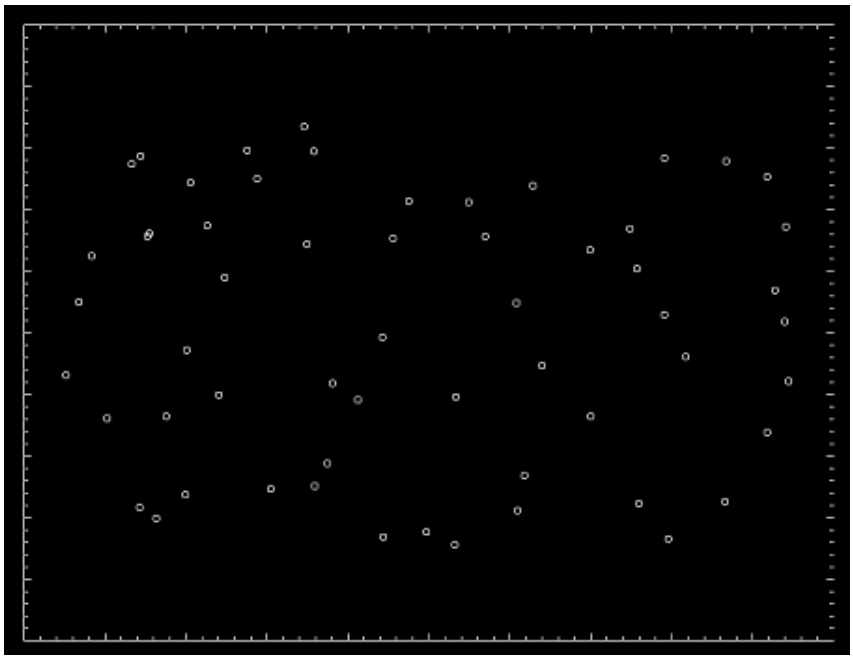
Nous voulons tracer une courbe d’approximation d’un nuage de points marqués à la souris dans l’écran, puis tracer sur cette courbe des lignes droites en ’épines’.

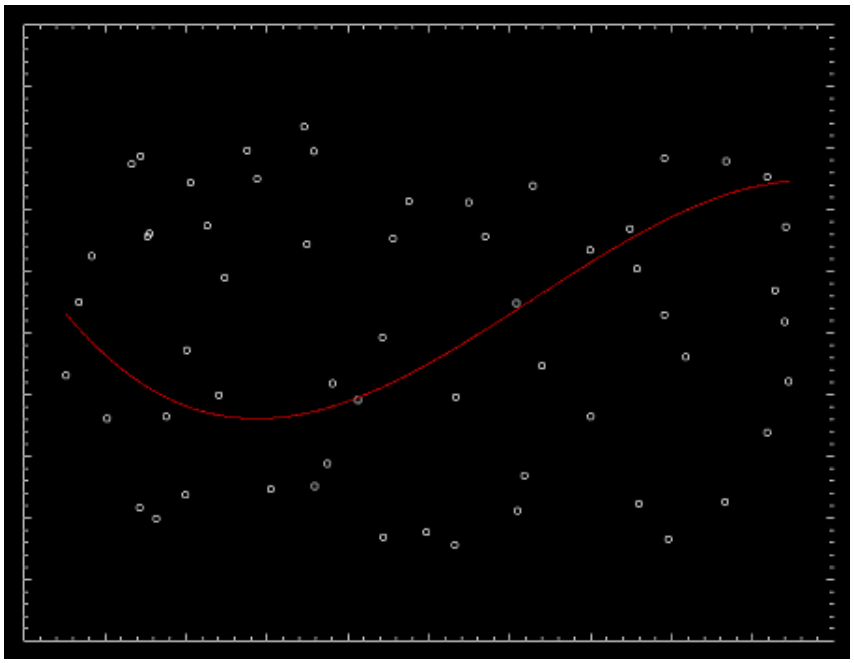
Les *épines de porc-épic* sont des lignes que l’on utilise pour détecter des points d’inflexion, des parties ’plates’ de la courbe, et des discontinuités dans sa courbure. C’est ce qui rend leur usage important en "Géométrie Computationnelle".

Nous avons développé un code pour calculer une courbe d’un degré? donné (entre 1 et 4) par la méthode des moindres carrés appliquée à^ un ensemble de points désignés par l’utilisateur dans l’interface graphique grâce à la souris. Il donne la possibilité d’ajouter des ’épines’ (avec une fréquence ajustable par l’utilisateur) sur la courbe afin de visualiser sa courbure.

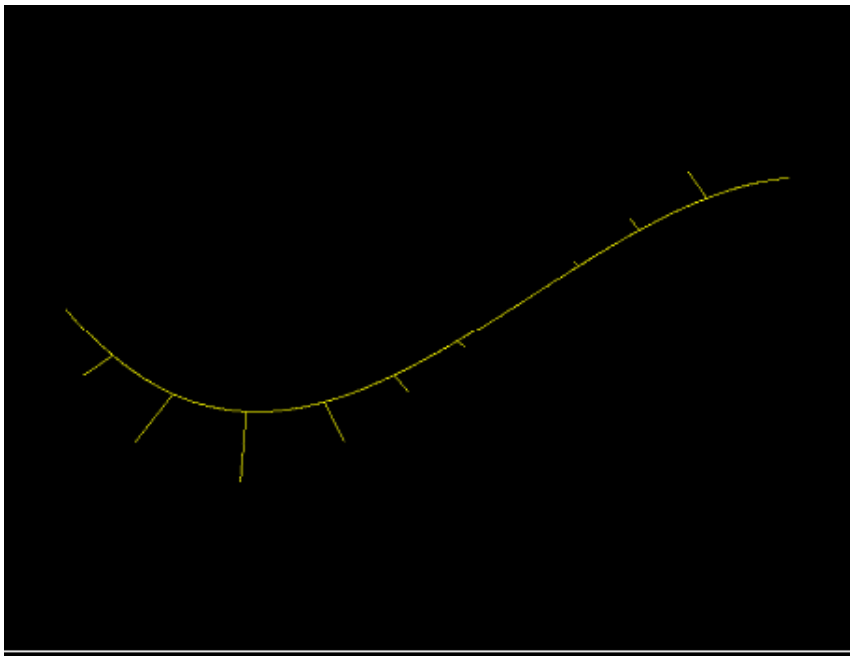
Sans entrer dans les détails de ce code (vous le trouverez à la fin de l’article) voici les résultats graphiques obtenus.

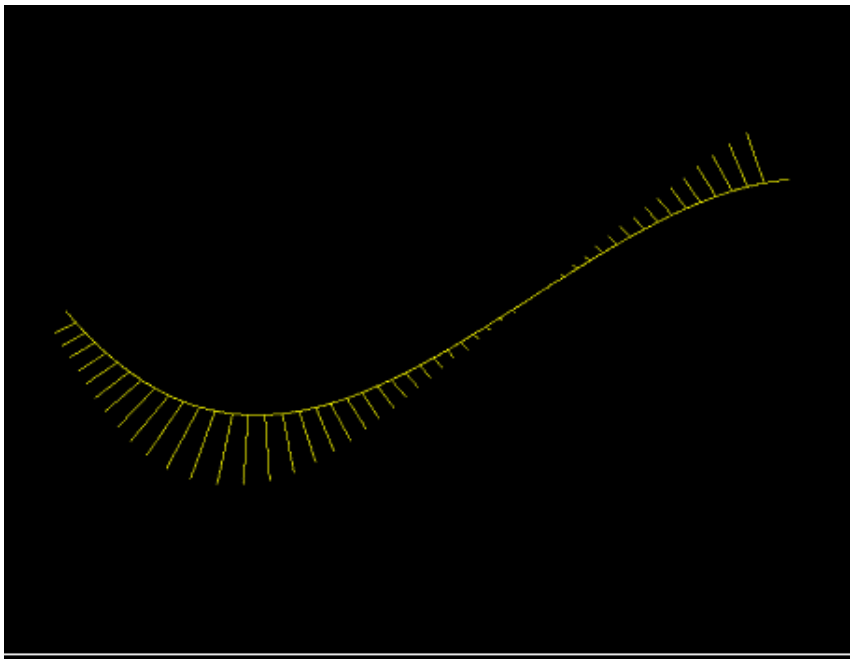
D’abord nous marquons les points à la souris et le code stocke leurs coordonnées dans un tableau. Puis il calcule la courbe du degré demandé? (entre 1 et 4) par l’utilisateur.



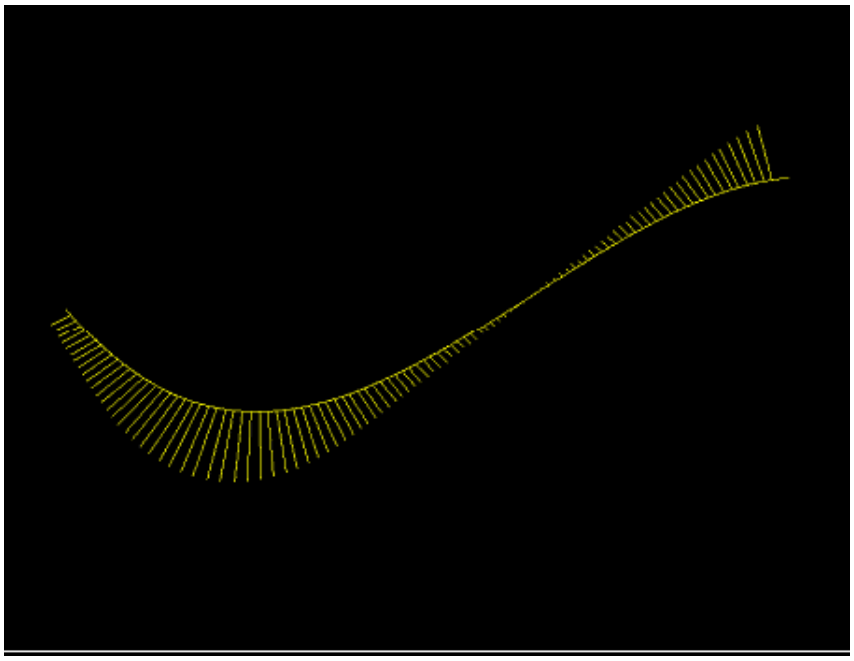


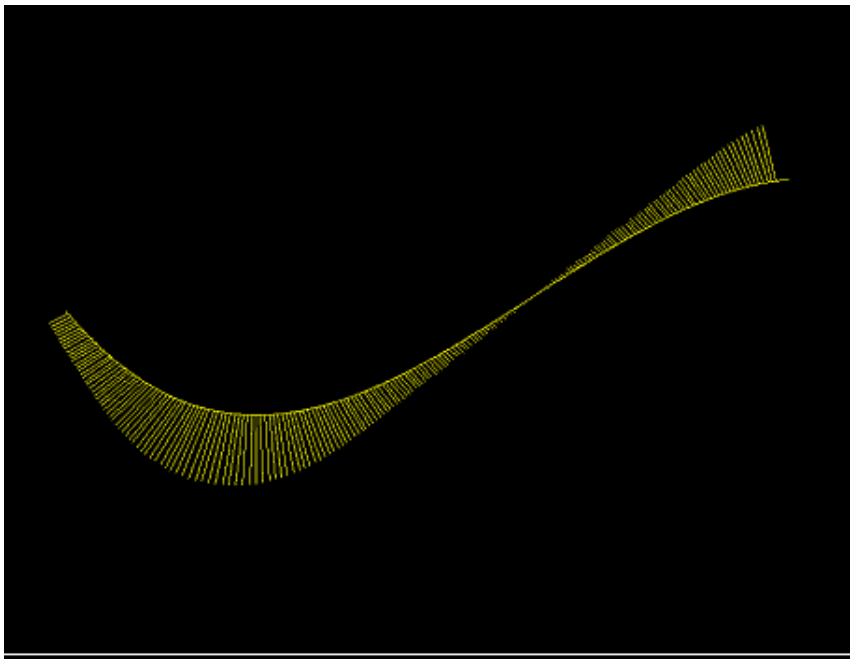
On trace alors les épines sur la courbe. Leur nombre peut être donné par l’utilisateur et le nouvel ensemble est tracé sur la même courbe.





La fréquence des épines peut être augmentée comme montré ci-dessous. Le programme peut aussi tracer une nouvelle courbe sur le même ensemble de données.





# Conclusion

Nous avons voulu montrer qu'il est possible d'installer une bibliothèque graphique interactive, PGPLOT, sous Linux et combien l'étendue des possibilités offertes est grande. Cet ensemble de routines vous apporte une réelle indépendance et vous fera gagner du temps si un affichage de résultats est nécessaire à chaque exécution de votre code. Inclure ces routines dans votre code en rend l'exploitation robuste et rapide.

# Références


- La page officielle PGPLOT : PGPLOT
- Code source de l'application 1):circle.f
- Code source de l'application 2):porcup.f
- Rapport technique sur l'application 2):601_2.pdf
- La page du cours de "Géométrie Computationnelle" Dr.Serdar CELEBI