

# Grafico di funzioni

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <crout.h> /* include le librerie grafiche */
#define npts 100 /* numero di punti calcolati per il grafico*/
/* Definizione della funzione */
double funzione(double x){
    ████████████████████ ;
    return( ██);
}
/* Routine per disegnare la funzione (passaggio di una funzione come
parametro) */
void plotfun(double fun(double),double xmin, double xmax){
    double xx[npts],yy[npts];
    int i;
    for(i=0;i<npts;i++){
        xx[i] =xmin+i/((double)npts)*(xmax-xmin);
        yy[i] = fun(xx[i]);
    }
    R_Graph(npts,xx,yy,"C");
}
```

```
int main(){
```

```
double xmin= ■ ;
```

```
double xmax= ■ ;
```

```
double ymin= ■ ;
```

```
double ymax=■ ;
```

limiti del grafico, ymin e ymax  
possono essere calcolati

```
/* Grafico di una funzione */
```

```
R_Init();          /* Apre la finestra grafica*/
```

```
R_SetGrid(1);     /* Opzione griglia sugli assi*/
```

```
R_DrawFrame(xmin,xmax,ymin,ymax); /* Disegna gli assi */
```

```
R_AxisTitles("Xaxis","Yaxis"); /* Scrive i nome degli assi */
```

```
plotfun(funzione,xmin,xmax); /* Disegna la funzione */
```

```
R_End();          /* Chiude la finestra */
```

```
}
```

```
#define npts 100
#define npar 2

/* Funzione di Fit (in questo caso una retta) */
double xfun(double x, int npar, double *par){
    double val;
    val = x*par[1]+par[0];
    return val;
}

/*.. sua derivata... */
double xder(double x, int npar, double *par){
    double val;
    val = par[1];
    return val;
}
```

```

// Calcolo della funzione da minimizzare (chi-quadro)
double xfcn(int npar, double *par){
    int    i;
    double  chi,chid, f=0;
    for (i=0; i < ndat; i++){
        chi  = (y[i] - xfun(x[i],npar,par));
        chid = pow(ey[i],2) + pow(xder(x[i],npar,par)*ex[i],2);
        f    += chi*chi/chid;
    }
    return f;
}

```

Infatti la formula del chi-quadro nel caso che non si trascurino gli errori sulle x è la seguente:

$$\chi^2 = \sum_i \frac{(y_i - f(x_i, p_1, \dots, p_M))^2}{\sigma y_i^2 + \sum_j \left( \frac{\partial f}{\partial x_j} \right)^2 \sigma x_j^2}$$

```
/* Routine per disegnare la funzione */  
void plotfun(double fun(double, int, double *), int npar,  
double *par, double xmin, double xmax){  
    double xx[npts],yy[npts];  
    int i;  
    for(i=0;i<npts;i++){  
        xx[i] = i/((double)npts)*(xmax-xmin)+xmin;  
        yy[i] = fun(xx[i],npar, par);  
    }  
    R_Graph(npts,xx,yy,"C");  
}
```

```

int main() {
    double par[npar]={0.0,1.0},outpar[npar],err[npar];
    double chiq;
    double xmin,xmax,ymin,ymax;
    int ndat;
    double *x,*y,*ex,*ey;
    /* Qui va inserita la lettura dei dati da file con allocazione
    dinamica*/
    /* Minimizzazione */
    R_MinuitSimpl(xfcn,npar,par,outpar,err,&chiq);
    /* Grafico */
    R_Init();           /* Apre la finestra grafica*/
    R_SetGrid(1);      /* Opzione griglia sugli assi */
    R_DrawFrame(xmin,xmax,ymin,ymax); /* Disegna gli assi*/
    R_AxisTitles("Xaxis","Yaxis");   /* Scrive i nome degli assi*/
    R_GraphErrors(ndat,x,y,ex,ey,"P"); /* Disegna punti con
    errore*/
    /* Disegna la funzione fittata n*/
    plotfun(xfun,npar,outpar,x[0],x[ndat-1]);
    R_End();           /* Chiude la finestra */
    return 0;
}

```