Librerie preconfezionate

- Sono routine già scritte da altri, di solito ben collaudate
- Comprendono le definizioni delle funzioni e il codice eseguibile
- Per librerie di pubblico dominio è disponibile anche il sorgente
- Richiedono l'inclusione di math.h all'interno del file sorgente
- È necessario linkare con opportune librerie extra -lm -lblas
- Bisogna leggere molto attentamente la documentazione

Un esempio: le librerie gsl

- GNU Scientific Library (GNU's Not Unix)
- Includono gsl.h o file analoghi
- Link sempre -lgsl e spesso -lgslcblas

Ma perché allora ho seguito questo corso?

- Per guidare l'auto bisogna saper almeno cambiare una ruota

- Un programma che metta insieme cose sconosciute è facilmente sbagliato

- Un algoritmo standard non va bene per la maggior parte delle applicazioni; i risultati originali hanno bisogno di programmi originali

Cosa si fa con le librerie gsl

- Valori di costanti matematiche

  $$\mathrm{M\_PI} = \pi$$

- Calcolo di funzioni elementari e funzioni speciali

  $$\mathrm{hypot}(3,4) \quad \mathrm{gsl\_pow\_4}(2)$$

- Valutazione di polinomi
  $p(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}$
  $\mathrm{gsl\_poly\_eval}(c, n, x)$

- Ordinamento di un vettore

  $$\mathrm{gsl\_sort}(v, k, n) \quad v_0, v_k, v_{2k}, \ldots$$

Inclusione e linking

Oltre ai due file che vanno inclusi sempre, ogni categorie di routine riochiede l'inclusione di altri file, ad esempio

#include ⟨gsl/gsl_xxx.h⟩
xxx = math, poly, sort_vector

Il linking viene fatto collegando libgsl e, quasi sempre, libgslcblas

gcc -lgsl -lgslcblas programma.c

BLAS = Basic Linear Algebra Subroutines è una collezione di funzione che permette di fare in modo ottimale alcune operazioni su vettori e matrici, come il prodotto di un vettore per uno scalare o la somma di due vettori

Strutture dati, argomenti delle funzioni e workspace

- Le routine scritte da altri utilizzano strutture dati proprie: il programmatore deve adattarsi

- Le funzioni utilizzano parametri di chiamata propri e parametri di uscita per verificare esattezza e precisione del risultato

- Alle funzioni serve anche uno spazio di lavoro temporaneo per velocizzare il programma: questo spazio deve essere allocato dall'utente

Esempio: integrazione con quadratura gaussiana

gsl_integration_qag(&F, a,b,epsabs,epsrel,limit,key, workspace, & result, & abserror)

- F è una struttura che contiene la funzione integranda
- a e b sono gli estremi di integrazione
- epsabs è l'errore assoluto richiesto, epsrel l'errore relativo richiesto
- key è collegato al numero di punti di integrazione gaussiana (va da 1 a 6 con numero crescente)

- L'algoritmo consiste nel dividere l'intervallo iniziale in sotto intervalli, i cui estremi sono memorizzati nel workspace. il massimo numero di intervalli è limit

- Workspace è una struttura di tipo gsl_integration_workspace e ha proprie funzioni di allocazione e deallocazione

- F è una struttura gsl_function che ha due membri: il primo è la funzione integranda f, il secondo un array di parametri da passare alla funzione f oltre a quello da integrare