Universitat de Barcelona

Facultat de Matemàtiques i Informàtica

Pràctica de Programació Científica

Simulació de trajectòries en camps de força central

Assignatura: Programació Científica
Curs: 2024–2025
Data: 20 de maig de 2025

Objectius

Aquesta pràctica té com a objectiu implementar una simulació del moviment d'un cos puntual sotmès a una força central conservativa en dues dimensions. L'estudiant aprendrà a:

- Utilitzar `structs` per emmagatzemar l'estat d'un sistema físic.
- Gestionar memòria dinàmica per emmagatzemar trajectòries temporals.
- Aplicar mètodes d'integració numèrica per resoldre equacions de moviment.
- Escriure i llegir fitxers amb condicions inicials i resultats.
- Visualitzar dades amb Gnuplot.

Marc teòric

El cos puntual es mou sota l'acció d'una força central de la forma:

$$\vec{F}(\vec{r}) = -k\frac{\vec{r}}{|\vec{r}|^3}$$

on $\vec{r} = (x, y)$ ès la posició respecte l'origen, i $k > 0$ ès una constant positiva.

Aquest tipus de forçes generen trajectòries tìpiques com cercles, el·llispes, paràboles i hipèrboles, segons l'energia total del sistema:

$$E = T + V = \frac{1}{2}mv^2 - \frac{km}{|\vec{r}|}$$

La simulació es realitzarà utilitzant dos mètodes d'integració:

- Euler (simple, menys precís, ès a implementar)
- Runge-Kutta d'ordre 4 (més precís, proporcionat)

Estructura de dades

Tots els estats del sistema es representaran amb la següent estructura:

```
typedef struct {
    char nom[100];
    double x, y;
    double vx, vy;
    double massa;
} Cos;
```

Aquesta estructura representa l'estat d'un cos puntual que es mou en el pla sota una força central. Conté la informació física necessària per descriure la seva evolució temporal:

- `char* nom`: Nom identificador del cos (per exemple, "Terra" o "Sonda1").
- `double massa`: Massa del cos, necessària per aplicar la segona llei de Newton.
- `double x, y`: Coordenades de la posició del cos en el pla.
- `double vx, vy`: Components de la velocitat en les direccions x i y.

Aquest conjunt de valors permet calcular la força sobre el cos i la seva energia, així com la seva trajectòria al llarg del temps mitjançant un mètode d'integració numèrica.

Enunciat detallat del projecte

Aquest projecte té com a objectiu simular el moviment de diversos cossos puntuals en el pla, sotmesos a una força central conservativa. El programa, escrit íntegrament en C, haurà de llegir les condicions inicials des d'un fitxer, calcular la trajectòria dels cossos mitjançant dos mètodes d'integració numèrica (Euler i Runge-Kutta d'ordre 4), i escriure els resultats en fitxers diferenciats per a cada mètode.

Lectura d'entrada

El programa ha de llegir des del fitxer `input.dad` les condicions inicials per a cada cos a simular. Aquest fitxer contindrà una línia per cada cos amb el següent format:

```
nom  x0  y0  vx0  vy0  massa
```

On:

- `nom` és una cadena de text que identifica el cos (sense espais).
- `x0`, `y0` són les coordenades inicials de posició.
- `vx0`, `vy0` són les components inicials de la velocitat.
- `massa` és la massa del cos en unitats arbitràries.

El programa ha de ser capaç de llegir múltiples cossos en una mateixa execució, emmagatzemant la seva informació inicial de forma estructurada mitjançant una `struct` definida a l'arxiu `main.c`.

Constants predefinides

Per simplificar l'ús del programa i evitar la introducció manual de paràmetres per part de l'usuari, s'utilitzaran constants definides al principi del fitxer `main.c` mitjançant directrius del preprocessador `#define`. Aquestes constants controlen el comportament global de la simulació i els seus valors inicials seran:

```
#define DT 0.01        // Pas de temps
#define T_MAX 10.0     // Temps total de simulació
#define K 1.0          // Constant de la força central
```

Amb aquests valors, la simulació integrarà el moviment dels cossos des del temps $t = 0$ fins a $t = T_{\text{MAX}}$, avançant en intervals de mida DT. La força central considerada és del tipus gravitatori o electrostàtic:

$$\vec{F}(\vec{r}) = -k\frac{\vec{r}}{|\vec{r}|^3}$$

Estructura i emmagatzematge de trajectòries

Per representar l'estat d'un cos en cada instant de temps, s'utilitzarà una estructura de tipus `Cos` amb els següents camps:

```
typedef struct {
    char nom[20];   // Nom identificador del cos
    double massa;   // Massa del cos
    double x, y;    // Posició en el pla
    double vx, vy;  // Velocitat en el pla
} Cos;
```

Per emmagatzemar les trajectòries al llarg de tota la simulació, el programa ha de reservar memòria dinàmica mitjançant la funció `malloc`. Concretament, per cada cos i per cada mètode (Euler i RK4), es reservarà un vector de `Cos` amb tants elements com passos de temps tingui la simulació. Aquestes estructures permetran reconstruir la trajectòria completa del cos, pas a pas, i escriure-la al fitxer de sortida.

Sortida

Per a cada cos llegit del fitxer d'entrada, s'han de generar automàticament dos fitxers de sortida:

- `nom_euler.res`
- `nom_rk4.res`

Aquests fitxers contindran, per cada pas de temps, l'estat del cos i l'energia mecànica total, amb el format següent:

```
t  x  y  vx  vy  energia
```

L'energia s'ha de calcular com la suma de l'energia cinètica i l'energia potencial generada per la força central. En cada pas:

$$E = \frac{1}{2}m(v_x^2 + v_y^2) - \frac{km}{\sqrt{x^2 + y^2}}$$

Aquest càlcul permetrà observar la conservació de l'energia i avaluar la qualitat numèrica de cada mètode d'integració.

Descripció del `main`

El programa principal (`main`) ha de seguir els següents passos de manera clara i seqüencial. Primer, llegeix el fitxer `input.dad` i emmagatzema la informació inicial de cada cos en una estructura de dades. A continuació, per a cada cos, inicialitza la seva trajectòria per separat amb els dos mètodes numèrics (Euler i RK4), copiant les condicions inicials al primer element de cada vector. El bucle principal del programa s'encarrega de simular pas a pas el moviment dels cossos, aplicant la funció corresponent a cada mètode, i actualitzant els vectors amb els nous estats.

Finalitzada la simulació, el programa escriu el contingut de cada trajectòria en els fitxers de sortida indicats, generant dos fitxers per cada cos. Finalment, allibera la memòria dinàmica i mostra per pantalla algun indicador com l'error màxim entre els dos mètodes o la confirmació de la finalització de la simulació per cada cos.

Tot el codi del projecte s'ha de desenvolupar en un únic fitxer `main.c`, evitant l'ús de múltiples mòduls per facilitar la comprensió del flux complet del programa.

Funcions a desenvolupar

- **Funció que implementa un pas d'Euler**:

```c
void pas_euler(Cos* estat, double dt);
```

  Actualitza la posició i velocitat del cos a partir de la força central i del pas de temps. Aquest mètode numèric permet aproximar la solució d'una equació diferencial de la forma $\vec{r}'' = \vec{F}(\vec{r})/m$, discretitzant el temps en passos de mida dt.

Entrada: estat inicial del cos (x, y, vx, vy), massa m, pas de temps dt
Sortida: nou estat del cos després d'un pas de temps

Funció `pas_euler(estat, dt)`
Calcula la força central:
$r \leftarrow \sqrt{x^2 + y^2}$
$fx \leftarrow -k \cdot x/r^3$
$fy \leftarrow -k \cdot y/r^3$
Actualitza la velocitat:
$vx \leftarrow vx + (fx/m) \cdot dt$
$vy \leftarrow vy + (fy/m) \cdot dt$
Actualitza la posició:
$x \leftarrow x + vx \cdot dt$
$y \leftarrow y + vy \cdot dt$
FiFunció

- **Funció que calcula la força central**:

```
void forca_central(double x, double y, double m, double* fx, double* fy);
```

Calcula la força central aplicada a un cos en funció de la seva posició i massa.

- **Funció que calcula l'energia total**:

```
double energia_total(Cos estat);
```

Retorna l'energia mecànica total del cos.

- **Funció que escriu la trajectòria a fitxer**:

```
void escriure_trajectoria(const char* nom_base, Cos* trajectoria, int n);
```

Escriu les dades de posició, velocitat i energia de cada pas temporal en un fitxer de sortida.

Exemple d'ús

Un exemple de fitxer `input.dad`:

```
orbita1 1.0 0.0 0.0 1.0 1.0
orbita2 1.0 0.0 0.0 1.4 1.0
```

Això genera els fitxers `orbita1_euler.res`, `orbita1_rk4.res` i `orbita2_euler.res`, `orbita2_rk4.res`. Per visualitzar les dades amb Gnuplot:

```
plot "orbita1_euler.res" using 2:3 with lines title "Euler", \
     "orbita1_rk4.res" using 2:3 with lines title "RK4"

plot "orbita1_euler.res" using 1:6 with lines title "Energia Euler", \
     "orbita1_rk4.res" using 1:6 with lines title "Energia RK4"
```