

Capítulo 9

Ecuaciones diferenciales

En esta práctica vamos a estudiar métodos numérico para obtener la solución de ecuaciones diferenciales del tipo

$$y'(x) = f(x, y)$$

y

$$y''(x) = f(x, y, y')$$

en un intervalo $[x_0, x_N]$. El intervalo lo dividimos en un conjunto de puntos igualmente espaciados una distancia h , que es el paso de integración de la ecuación diferencial.

Estudiaremos casos simples de ecuaciones lineales de la Dinámica Newtoniana.

9.1. Método de Euler

El método de Euler consiste en aproximar la derivada por la fórmula de dos puntos

$$y'(x) \sim \frac{y(x+h) - y(x)}{h}$$

con lo que obtenemos la relación de recurrencia

$$y(x+h) = y(x) + hf(x, y)$$

Converge en orden $O(h)$, por lo que hay que tomar h muy pequeño para obtener resultados razonables.

9.2. Método del punto medio

El método del punto medio consiste en aproximar la derivada por la fórmula de dos puntos

$$y'(x) \sim \frac{y(x+h) - y(x-h)}{2h}$$

con lo que obtenemos la relación de recurrencia

$$y(x+h) = y(x-h) + 2hf(x, y(x))$$

Converge en orden $O(h^2)$, lo que es en general suficiente para muchas aplicaciones, aunque puede presentar inestabilidades. Esta relación no está definida en el primer punto del intervalo, $x_0 + h$, ya que implica el valor de y en $x_0 - h$. Para este primer punto se utiliza el método de Euler. Si denotamos $x_n = x_0 + nh$ e $y_n = y(x_n)$ podemos escribir

$$\begin{aligned} y_1 &= y_0 + hf(x_0, y_0) \\ y_{n+1} &= y_n + hf(x_n, y_n) \end{aligned}$$

9.3. Método de Runge-Kutta de 4 orden

Entre los métodos más utilizados están los de Runge-Kutta de orden 4 y superior. El método de Runge-Kutta de cuarto orden converge en orden h^4 . Consiste en la siguiente regla:

$$\begin{aligned} y_{n+1} &= y_n + \frac{h}{6} (k_1 + 2k_2 + 2k_3 + k_4) \\ k_1 &= f(x_n, y_n) \\ k_2 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1) \\ k_3 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2) \\ k_4 &= f(x_n + h, y_n + hk_3) \end{aligned} \tag{9.1}$$

Se reduce a la regla de Simpson cuando f no depende de y .

9.4. Ecuaciones de segundo orden

En el caso de una ecuación $y'' = f(x, y, y')$, reemplazamos la ecuación de segundo orden por dos ecuaciones de primer orden con el cambio de variables

$$\begin{aligned} y_1(x) &= y(x) \\ y_2(x) &= y'(x) \end{aligned}$$

con lo que queda el sistema de ecuaciones diferenciales de primer orden

$$\begin{aligned} y_1'(x) &= y_2(x) \\ y_2'(x) &= f(x, y_1(x), y_2(x)) \end{aligned}$$

que se pueden resolver simultáneamente por cualquier método válido para ecuaciones diferenciales de primer orden, entre ellos los tres que acabamos de ver.

9.5. Aplicación de los métodos a casos de interés físico

Vamos a utilizar los métodos de integración numérica en ecuaciones diferenciales sencillas que aparecen en dinámica. En una primera fase utilizaremos el método más simple, es decir el método de Euler. Después podéis utilizar cualquier método más sofisticado. En general el método del punto medio da resultados muy aceptables.

Los dos ejercicios que vamos a estudiar son:

1. Resolver la ecuación diferencial de una pelota que cae de una altura x_0 , partiendo del reposo. Hay que imponer las condiciones de rebote de que cuando la altura x sea menor que el paso de integración, se integra en esta fracción de paso de integración y la velocidad cambia de signo, es decir si el paso de integración da x_{new} y v_{new} en función de x_{old} y v_{old} , cuando hay rebote, $x_{new}=x_{old}$ y $v_{new}=-v_{old}$. Por lo tanto el problema, aunque sencillo, esconde una singularidad que hace que la velocidad cambie de signo. La ecuación diferencial a resolver es

$$m \frac{d^2x}{dt^2} = F$$

donde $F = -mg$. Esta ecuación se resuelve como un conjunto de dos ecuaciones diferenciales:

$$\begin{aligned} \frac{dx}{dt} &= v \\ m \frac{dv}{dt} &= F \end{aligned}$$

El programa `rebote_el_eu.cpp` contiene la programación para el caso elástico, por el método de Euler.

```
#include <iostream>
#include <iomanip>
#include <fstream>
using namespace std;
int main()
{
    const char *tab = "    ";
    double x0 = 1.;
    double m = 1.;
    double g = 9.81;
    double f = -m * g;
    double k = 0.;
    double x = x0;
    double v0 = 0.;
    double t = 0;
    double h = 0.01;
```

```

double xold = x0;
double vold = v0;
//Numero de pasos de integracion
int n = 1000;
double xnew, vnew;
ofstream fout("rebote_el_eu.res");
//Integracion
for (int i = 0; i < n; i++) {
xnew = xold + vold * h;
vnew = vold + (f + k * vold) / m * h;
//condicion de rebote;
//delth fraccion de h para que la pelota llegue al suelo
//La velocidad se invierte en el choque
if (xnew <= 0) {
    xnew=0;
        double deltah=-xold/vold;
        vnew = -(vold + (f + k * vold) / m * deltah);
        t = t + deltah; /*se desprecia el efecto de la aceleracion en el rebote */
} else
    t = t + h;
xold = xnew;
vold = vnew;
fout << setprecision(10) << setw(15) << t << tab <<
    setw(15) << xnew << tab << setw(15) << vnew <<
    endl;
}
fout.close();
}

```

El programa `rebote_el_pm.cpp` contiene la programación para el método del punto medio. Las condiciones iniciales son $x=x_0$ y $v=0$. Integraréis un número importante de pasos de integración, como *10000* ó *100000*, de forma que haya varios rebotes. El primer aspecto a estudiar es la estabilidad de la integración numérica. así, no os debe extrañar de que si el paso de integración es demasiado grande la pelota rebote cada vez más alto o más bajo debido a una ganancia o pérdida de energía producida por los errores de redondeo al propagarse. Estos problemas deben de disminuir y eventualmente desaparecer al disminuir el paso de integración. Como ejemplo de los resultados que se obtienen, la Fig. 1 da los resultados de integrar con el método de Euler y $h=0.01$ y la Fig.2 el resultado de integrar con el método del punto medio. En una segunda fase, cuando el rebote elástico funcione correctamente podéis incluir los siguientes aspectos:

- Una fuerza de rozamiento kv , donde k es una constante. La podéis elegir como la constante propia para una esfera o simplemente una cantidad numéricamente pequeña

Figura 9.1: Integración por el método de Euler

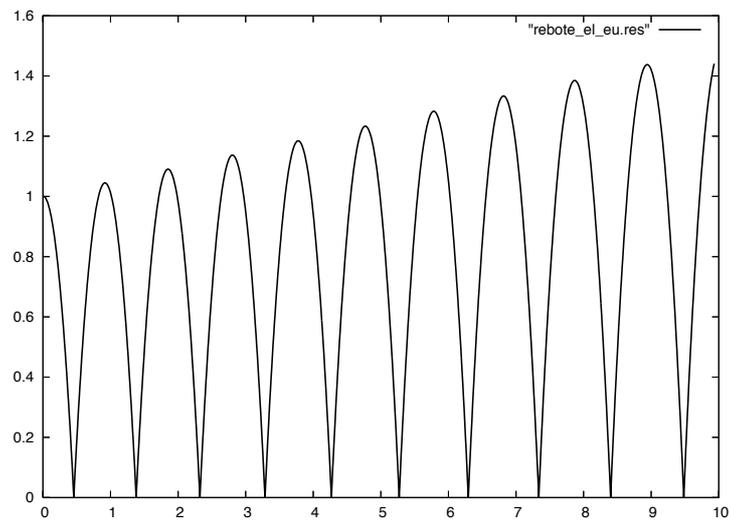
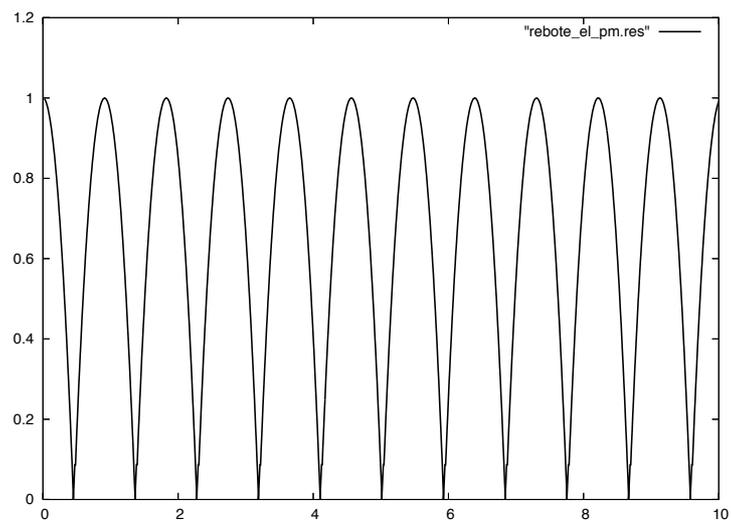


Figura 9.2: Integración por el método del punto medio



si no queréis repasar vuestros conocimientos de aerodinámica en un libro de física general.

- Un coeficiente de inelasticidad k de forma que en el rebote $v_{new} = -k v_{old}$.
- Finalmente podéis incluir un método de integración más eficiente y comparar. Como recordatorio os doy las ecuaciones en diferencias en el caso de Euler y del punto medio:

- Euler

$$x_{n+1} = x_n + v_n h$$

$$v_{n+1} = v_n - g h$$

$$t_{n+1} = t_n + h$$

- Punto medio

$$x_{n+1} = x_{n-1} + 2v_n h$$

$$v_{n+1} = v_{n-1} - 2g h$$

$$t_{n+1} = t_n + h$$

con

$$x_1 = x_0 + v_0 h = x_0$$

$$v_1 = v_0 - g h = -g h$$

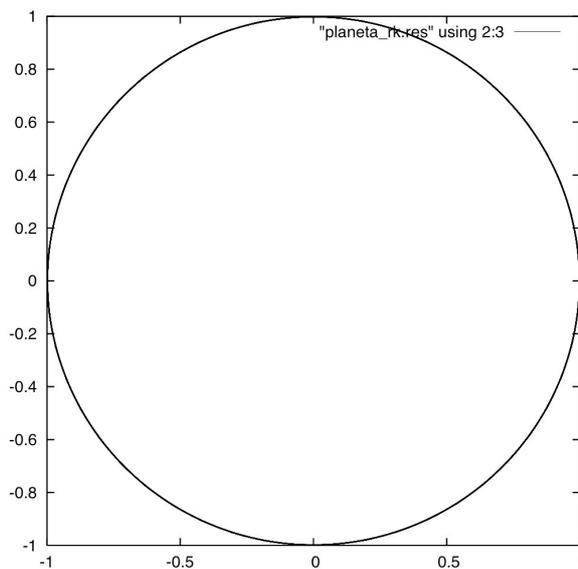
$$t_1 = h$$

2. Consideremos el movimiento de la Tierra alrededor del Sol. Integramos las ecuaciones de Newton de forma cartesiana. Verificad la estabilidad de la integración integrando varias órbitas consecutivas. Las ecuaciones del movimiento que hay que resolver son

$$m \frac{d^2 x}{dt^2} = F_x = -\frac{GMm}{r^3} x$$

$$m \frac{d^2 y}{dt^2} = F_y = -\frac{GMm}{r^3} y$$

Estas ecuaciones las escribiréis como cuatro ecuaciones diferenciales de primer orden, que resolveréis en primer lugar por el método de Euler y en segundo lugar por el método del punto medio. Tomad un paso de integración de $h=0.01$ e integrad 10000 pasos de integración lo que corresponde a 100 años en intervalos de tres días. Comparad el comportamiento de ambos métodos de integración. Comparad después con $h=0.001$. Para evitar números excesivamente grandes es conveniente utilizar unidades de longitud astronómicas: $1AU = 1.496 \times 10^{11} m$, que corresponde a la distancia promedio de la Tierra al Sol. En estas unidades el radio de la órbita vale la unidad en el caso de órbitas circulares. En estas unidades el producto de la masa de la Tierra por la constante gravitacional vale $GM = 4\pi^2 AU^3 / año^2$. Tomad como condiciones iniciales las correspondientes a órbitas



(a)

Figura 9.3: Integración de la órbita durante un siglo por el método de Runge-Kutta 4 orden

circulares $v_0 = (GM/R_0)^{\frac{1}{2}}$. En una segunda etapa considerad órbitas cuya velocidad inicial se desvía por encima y por debajo de la condición de órbitas circulares. Representad x en función de y . Discutid el resultado. Representad también v_x en función de v_y . Para visualizar la estabilidad del método de integración un buen indicador es el radio de la órbita en función del tiempo. En el caso de órbitas circulares debe de permanecer prácticamente constante integrando durante algunos siglos.

Para visualizar los resultados es conveniente que vuestro programa efectúe una salida en un fichero que se pueda visualizar fácilmente con *Gnuplot*. Para el primer ejercicio por ejemplo, podéis imprimir en un fichero t,x,v en forma de tres columnas. Si el fichero se llama *rebote.res* por ejemplo, el comando plot “*rebote.res*” de *Gnuplot* dibujara x en función de t . Con el comando *using* de *Gnuplot* podéis dibujar cualquier columna en función de cualquier otra. En el segundo ejercicio podéis sacar t,x,y,r (radio de la órbita), v_x,v_y .

Los programas `planeta_eu.cpp`, `planeta_pm.cpp`, `planeta_rk.cpp` resuelven el problema por los métodos de Euler, punto medio y Runge-Kutta, respectivamente. A continuación damos el programa `planeta_rk.cpp` :

```
#include <cmath>
#include <iomanip>
#include <fstream>
#include <iostream>
using namespace std;
void rk4(double[], double[], int, double, double, double[],
  void (*derivs) (double, double[], double[]));
void derivs(double, double[], double[]);
int main()
{
  int neq = 4; //numero de ecuaciones de primer orden
  char *tab = "          ";
  double GM = 4 * pow(M_PI, 2);
  double m = 5.99e24;
  double M = 1.99e30;
  double t = 0;
  double h = 0.02;
  double v0 = sqrt(GM);
  double T = sqrt(4 * pow(M_PI, 2) / GM);
  cout << "Periodo teórico= " << T << endl;
  double tprev = 0.;
  double xold = 1.;
  double vxold = 0;
  double yold = 0;
  double vyold = v0;
  double y[4] = { xold, vxold, yold, vyold };
  double yn[4];
  double k1[4];
  double r = sqrt(pow(xold, 2) + pow(yold, 2));
  int n = 10000;
  double xnew, vxnew, ynew, vynew;
  ofstream fout("planeta_rk.res");
  for (int i = 0; i < n; i++) {
    derivs(t, y, k1);
    rk4(y, k1, neq, t, h, yn, (*derivs));
    xnew = yn[0];
    ynew = yn[2];
    vxnew = yn[1];
    vynew = yn[3];
  }
}
```

```

//condicion de periodo;
if (ynew * yold < 0 && xnew > 0) {
    T = t - tprev;
    cout << "tprev= " << tprev << tab << "t= " << t
<< tab << "Periodo= " << T << endl;
    tprev = t;
}
y[0] = yn[0];
y[1] = yn[1];
y[2] = yn[2];
y[3] = yn[3];
t = t + h;
r = sqrt(pow(xnew, 2) + pow(ynew, 2));
fout << t << tab << xnew << tab << ynew << tab <<
    vxnew << tab << vynew << tab << r << endl;
}
}
void rk4(double y[], double k1[], int n, double x, double h,
    double yout[], void (*derivs) (double, double[],
double[]))
{
    double xh, hh, h6;
    double *k4 = new double[n];
    double *k23 = new double[n];
    double *yt = new double[n];
    hh = h * 0.5;
    h6 = h / 6.0;
    xh = x + hh;
    for (int i = 0; i < n; i++)
yt[i] = y[i] + hh * k1[i];
    (*derivs) (xh, yt, k4);
    for (int i = 0; i < n; i++)
yt[i] = y[i] + hh * k4[i];
    (*derivs) (xh, yt, k23);
    for (int i = 0; i < n; i++) {
yt[i] = y[i] + h * k23[i];
k23[i] += k4[i];
    }
    (*derivs) (x + h, yt, k4);
    for (int i = 0; i < n; i++)
yout[i] =
    y[i] + h6 * (k1[i] + k4[i] + 2.0 * k23[i]);
    delete[]yt;
}

```

```

    delete[]k23;
    delete[]k4;
}
void derivs(double x, double y[], double dy[])
{
    double GM = 4 * pow(M_PI, 2);
    double r = sqrt(y[0] * y[0] + y[2] * y[2]);
    double r3 = r * r * r;
    dy[0] = y[1];
    dy[1] = -GM / r3 * y[0];
    dy[2] = y[3];
    dy[3] = -GM / r3 * y[2];
}

```

9.6. Ejercicios a presentar como memoria.

1. Estudiar para que paso de integración no hay cambio apreciable de altura en 10 rebotes consecutivos con el método de Euler.
2. Estudiar para que paso de integración el método del punto medio tiene errores de integración visibles (cambio de altura en el rebote) en 10 rebotes consecutivos. Ajustad el número total de pasos cada vez que cambiéis el paso de integración para tener siempre unos 10 rebotes.
3. Introducir con el método del punto medio un coeficiente de inelasticidad, en el que se pierda el 10 % de la energía cinética en el rebote.
4. Estudiar la estabilidad con el paso de integración h en el problema del planeta, para los métodos del punto medio y RK4. Aumentad el paso de integración hasta observar que el radio de órbitas circulares varía con el tiempo (debido a errores numéricos).
5. Modificad la velocidad inicial en `planeta_rk.cpp` para obtener órbitas elípticas. Dibujadlas con Gnuplot. Utilizad la opción `set size square` para que las escalas en los ejes x e y sean las mismas y no haya distorsiones debidas a las escalas.
6. Obtener la solución de un oscilador anarmónico forzado con una fuerza sinusoidal por el método del punto medio y RK4:

$$\frac{d^2x}{dt^2} + k^2x + \beta x^3 = A \sin(\omega_0 t)$$

Elegid las constantes k , β , A , ω_0 simples, por ejemplo podeis comenzar por 1, 1, 0.1, 1 (unidades MKS). Tomad como condiciones iniciales $\dot{x} = 0$ m/s y $x_0 = 1$ m.