

Introducción a la solución numérica de ODE's

Guía computacional 1 - Mecánica Clásica 2016 - Clase G. Mindlin

Ignacio Poggi - L.U: 567/07 - ignaciop.3@gmail.com

29 de marzo de 2016

1. Enunciado

En la sección de materiales adicionales de la cátedra se encuentra un programa principal y un integrador Runge-Kutta de orden 4 (rk4) en lenguaje C. En el programa principal se encuentra escrita una ecuación diferencial a integrar, los parámetros y las condiciones iniciales. Sobre este código van a trabajar en las siguientes actividades realizando las modificaciones pertinentes para su problema en particular.

En ubuntu es posible compilar y ejecutar el código directamente desde una terminal abierta en una carpeta que contenga tanto el programa principal como el integrador rk4:

```
gcc ODE_ejX.c -o ode_ejX -lm rk4.c
```

```
./ode_ejX
```

Se obtendrá como salida un archivo llamado *ejX.dat*, donde *X* es el número del ejercicio, que contiene el resultado de la integración. Los resultados pueden ser analizados graficamente mediante un graficador, en nuestro caso utilizaremos **gnuplot** que se controla mediante comandos en terminal.

Actividad 1

Editar el código de ODE.c para analizar los siguientes puntos:

- Cómo varía el resultado según el paso de integración. Programe una integración con el método de Euler y compare.
- Analizar cómo evoluciona el sistema dadas distintas condiciones iniciales.

Qué tipo de conclusiones puede obtener a partir de los análisis anteriormente realizados.

Actividad 2 - Oscilador armónico amortiguado

El oscilador armónico amortiguado es un problema del cual se conoce la solución analítica cuya ecuación diferencial que rige el movimiento es:

$$\frac{d^2x}{dt^2} + 2\gamma \frac{dx}{dt} + \omega_0^2 x = 0 \quad (1)$$

Estudie numéricamente las soluciones del sistema según la relación de los parámetros, para ello: escriba la ecuación de segundo orden como dos ecuaciones de primer orden, varíe γ y ω e integre. También analice distintas condiciones iniciales. Compare con lo conocido de la solución analítica, para ello grafique como evoluciona la posición en el tiempo, la velocidad y cuál es la trayectoria en el espacio de fases $x\dot{x}$.

Actividad 3 - Oscilador de Van der Pol

Es un tipo de oscilador con un amortiguamiento no lineal descrito a principio de siglo por Van der Pol quien estudió circuitos eléctricos con componentes no lineales obteniendo la ecuación:

$$\frac{d^2x}{dt^2} + \mu(x^2 - 1)\frac{dx}{dt} + x = 0 \quad (2)$$

Este sistema presenta soluciones oscilatorias para ciertos valores del parámetro μ que son conocidas como oscilaciones de relajación. Esta ecuación tiene una importancia en la ciencia ya que fue usada en distintos campos para describir por ejemplo, el comportamiento de una falla tectónica o el potencial de acción de una neurona. Esto se debe a que el sistema según los valores de x presenta un amortiguamiento positivo (como el de la actividad 2 donde el sistema pierde energía), y para otros presenta un amortiguamiento "negativo" donde el sistema gana energía. Esto produce que eventualmente la energía perdida en un ciclo sea igual a la ganada generando oscilaciones autosostenidas. Este sistema se verá con más detalle avanzado el curso, en esta práctica se propone realizar un acercamiento de forma numérica para tener cierta comprensión de cómo se comporta el mismo.

- Escriba el sistema como dos ecuaciones de primer orden.
- Inspeccione numéricamente las soluciones posibles del sistema, estudie como varían según la variación del parámetro μ . Para ello grafique la trayectoria x en función del tiempo, la velocidad \dot{x} en función del tiempo y también el espacio de fases $x\dot{x}$.
- Modifique también las condiciones iniciales y estudie numéricamente las respuestas del sistema. Para ello grafique la trayectoria x en función del tiempo, la velocidad \dot{x} en función del tiempo y también el espacio de fases $x\dot{x}$.

A entregar

Se deberá entregar un trabajo de la actividad 2 y 3, con los códigos, los gráficos obtenidos para las integraciones numéricas propuestas y el correspondiente análisis para cada caso.

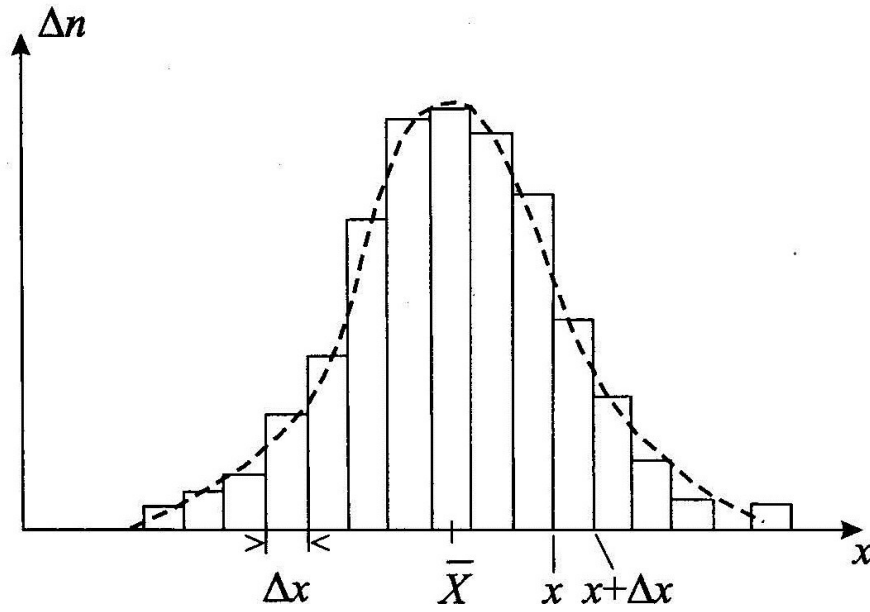


Figura 1: Histograma

2. Análisis de datos y conclusiones

2.1. Oscilador armónico amortiguado

El oscilador armónico amortiguado es el caso generalizado de los osciladores armónicos libres ya que su comportamiento contempla la disipación de energía pero no la presencia de fuerzas externas. Su comportamiento está dado por (1). Para obtener las soluciones de dicha ecuación, se propone una solución de la forma $x(t) = e^{zt}$, con $z \in \mathbb{C}$. Reemplazando el $x(t)$ propuesto en (1), se obtiene la siguiente ecuación cuadrática para z :

2.2. Oscilador de Van der Pol

A partir de las 50 mediciones del periodo del pendulo hechas en cada serie a distintas longitudes, se calculó su valor medio.

Es importante aclarar que obtenidos los valores medios fue necesario conocer la desviación de cada uno de ellos. Para lo cual aplicamos la siguiente fórmula:

$$\Delta y^2 = \sum_{k=1}^n \left(\frac{\partial f}{\partial x_i}(x_1, x_2, \dots, x_n) \cdot \Delta x_i \right)^2 \quad (3)$$

ΔT fue obtenido a partir del software Origin. Mientras que para ΔL tuvimos que aplicar la

fórmula (3) ya que la medición del punto de vínculo al centro de masa lo hicimos tomando las longitudes del hilo, la cabeza del tornillo y el radio de la esfera.

A partir de los datos obtenidos en la tabla (??) de forma directa, se realizó un test de hipótesis estadístico. Es decir, determinar si los valores corresponden a una distribución gaussiana. Para ello, tomamos diferentes mediciones del periodo a distintas longitudes y graficamos la información recolectada mediante histogramas. Con esta información se procedió a un ajuste Gaussiano

He aquí las distintas situaciones.

3. Apéndice

Descargar el archivo *guia1_poggi.zip* y descomprimirlo en una carpeta a elección del usuario, abrir una terminal dentro de la carpeta $\sim \backslash mecanica-clasica \backslash practica1$; y seguir las instrucciones provistas en la sección Enunciado.

Los archivos que contienen los datos numéricos (*ej2.dat* y *ej3.dat*) no fueron transcritos en este informe dada la extensión de los mismos

3.1. Código fuente en C del oscilador armónico amortiguado (ODE_ej2.c)

```
#include <stdio.h>
#include <math.h>

#define a -1

// Define parametros para usar en el sistema en todo el codigo
struct Par{
    double gamma, omega;
} aa;

// Ecuaciones del sistema
void ecuaciones(int n, double v[], double dv[], double t){
    double x,y;
    x=v[0];
    y=v[1];

    // En este caso son 2 ecuaciones acopladas
    dv[0]= y;
    dv[1]= -2*aa.gamma*y-aa.omega*aa.omega*x;

    return;
}
```

```

// Programa principal
int main() {

    int i, j;
    FILE *ptr;
    double v[2], t, dt, t_pre, t_max;

    // Archivo de salida
    ptr=fopen("ej2.dat", "w");

    dt=0.01;
    t_max=2;

    // Condiciones iniciales. Se analizaron 3 casos: sub, sobre y
    // amortiguamiento critico variando gamma y omega con valores
    // opuestos en el rango de 1 a 10, con pasos de 5
    for (j=1; j<=10; j=j+4) {
        v[0] = 0.5*j;
        v[1] = 0.5*(10-j);
        aa.gamma=(10-j);
        aa.omega=j;

        t=0.;

        while(t<t_max) {
            // Integra las ecuaciones utilizando el metodo de Runge
            // Kutta
            rk4(ecuaciones, v, 2, t, dt);

            // Imprime la integracion
            fprintf(ptr, "%lg\t%lg\t%lg\n", t, v[0], v[1]);

            t+=dt;
        }

        fprintf(ptr, "\n");
    }

    fclose(ptr);
    return(0);
}

```

3.2. Código fuente en C del oscilador de Van der Pol (ODE_ej3.c)

aasdasdjkasdj

3.3. Código fuente en C del método de Runge-Kutta de orden 4 provisto por la cátedra (rk4.c)

```
/* Runge Kutta integrator from numerical recipies plus improvements */  
/* void *deri(int n,double h[],double D[],double t); */  
/* function argument not tested yet */
```

```
void rk4(void deri(int , double [], double [], double ), \  
double h[], int n, double t, double dt)  
{  
#define naux 26  
  
int i;  
double k1[naux],k2[naux],k3[naux],k4[naux],h0[naux];  
double dt2, dt6;  
  
dt2=dt/2.;  
dt6=dt/6.;  
  
for (i = 0 ; i<n; i++)  
    h0[i] = h[i];  
  
deri(n,h0,k1,t);  
for (i =0 ; i<n ; i++)  
    h0[i]=h[i]+dt2*k1[i];  
  
deri(n,h0,k2,t+dt2);  
for (i =0 ; i<n ; i++)  
    h0[i]=h[i]+dt2*k2[i];  
  
deri(n,h0,k3,t+dt2);  
for (i =0 ; i<n ; i++)  
    h0[i]=h[i]+dt*k3[i];  
  
deri(n,h0,k4,t+dt);  
for (i = 0 ; i<n ; i++)  
    {h0[i]=h[i]+dt*k4[i];};  
  
for (i =0; i<n ; i++)  
    h[i]=h[i]+dt6*(2.*(k2[i]+k3[i])+k1[i]+k4[i]);
```

```
return ;  
}
```

4. Bibliografía