

Métodos Computacionales

Taller 5 — 2018-10

La solución a este taller debe subirse por SICUA antes de las 5:00PM del lunes 14 de mayo del 2018. Si se entrega la tarea antes del lunes 23 de abril del 2018 a las 11:59PM los ejercicios se van a calificar con el bono indicado.

Los archivos del código deben estar en un único repositorio `NombreApellido_taller5`, por ejemplo si su nombre es Malena Pichot el repositorio debe llamarse `MalenaPichot_taller5`. Al clonarlo debe crearse la carpeta `MalenaPichot_taller5` con tres carpetas: `punto_1`, `punto_2` y `punto_3`.

En la implementación principal de los algoritmos solicitados la copia y reutilización de código de cualquier fuente de internet (incluido el repositorio del curso) deja la nota en cero.

Todas las respuestas deben ser escritas en C++ y cada carpeta debe incluir el makefile para compilar el código, ejecutarlo y producir la gráfica solicitada.

1. Caos

(30 (35) puntos) Escriba un código para resolver el siguiente sistema de ecuaciones con un método Runge-Kutta de cuarto orden.

$$\dot{q}_2 = p_1, \quad (1)$$

$$\dot{q}_2 = p_2, \quad (2)$$

$$\dot{p}_1 = -\frac{2q_1}{(4q_1^2 + \epsilon^2)^{3/2}}, \quad (3)$$

$$\dot{p}_2 = \frac{q_1 - q_2}{((q_1 - q_2)^2 + \epsilon^2/4)^{3/2}} - \frac{q_1 + q_2}{((q_1 + q_2)^2 + \epsilon^2/4)^{3/2}}. \quad (4)$$

Tome un paso de tiempo $\Delta t = 0.006$, un tiempo total de $t = 3000$, condiciones iniciales $(q_1, p_1) = (a, 0)$ y $(q_2, p_2) = (-a, 0)$ con $a = 1/(2\sqrt{2})$ y $\epsilon = 10^{-3}$.

El movimiento descrito por q_1 es periódico. Para visualizar el comportamiento de las variables, cada vez que q_1 pase de ser positivo a negativo el código va a imprimir en pantalla los valores de q_2 y p_2 . Luego un código de python debe preparar la gráfica de q_2 vs. p_2 .

El código fuente debe llamarse `caos.cpp`, el ejecutable `caos.x`, el script de python para hacer la gráfica `caos.py` y la gráfica final `caos.pdf`.

2. Condensador de placas paralelas

Considere un condensador de placas paralelas ubicado en una región bidimensional del espacio, de dimensiones $L \times L$. Las placas tienen un largo l y una separación d entre ellas, como se muestra en la figura 1.

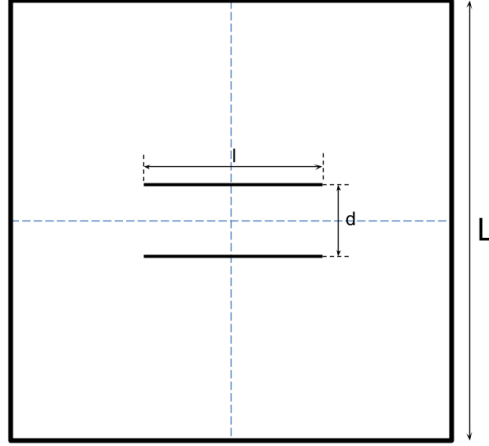


Figura 1: Condensador de placas paralelas.

Supongamos que existe una diferencia de potencial constante V_0 entre las placas (una de las placas se encuentra a $-V_0/2$ y la otra a $V_0/2$). Además, en el borde de la región tomemos el potencial fijo en 0.

El potencial eléctrico $V(x, y)$ en la región debe cumplir la ecuación de Laplace

$$\nabla^2 V(x, y) = \frac{\partial^2 V(x, y)}{\partial x^2} + \frac{\partial^2 V(x, y)}{\partial y^2} = 0. \quad (5)$$

Así mismo el campo eléctrico está dado por.

$$\mathbf{E}(x, y) = -\nabla V(x, y) = \left(-\frac{\partial V(x, y)}{\partial x}, -\frac{\partial V(x, y)}{\partial y} \right) \quad (6)$$

Para calcular el potencial numéricamente se debe discretizar la región como una matriz de tamaño $L/h \times L/h$, donde h es la separación entre los puntos y utilizar el esquema de diferencias finitas.

Escriba un programa en C++ (`placas.cpp`) que encuentre el potencial eléctrico con el método de relajación usando N iteraciones. El mismo programa debe calcular el campo eléctrico a partir de la solución final del potencial. Escriba un código en Python `grafica.py` que grafique el potencial y el campo eléctrico usando `imshow` y `streamplot` en una grafica llamada `placas.pdf`.

Utilice $L = 5$ cm, $l = 2$ cm, $d = 1$ cm, $h = 5/512$ cm, $V_0 = 100$ V y $N = 2 \times (L/h)^2$ iteraciones.

Nota: Utilice esta libreria <https://github.com/glennrp/libpng> para leer y escribir las imágenes. Para la transformada de Fourier debe escribir su propia implementación. En todos los casos el ejecutable debe poder crearse con el comando `make`.