

## 1. Descripción general:

El proyecto consiste en la resolución de la ecuación de Poisson a través de una implementación del método de diferencias finitas, haciendo uso de herramientas computacionales.

Las herramientas a utilizar son C, para crear un programa que luego será empaquetado en un módulo para Python. Una vez que se ha trasladado el proceso a Python, se utiliza VPython para la visualización de los resultados.

## 2. Sobre el método de diferencias finitas:

### 2.1. 1 variable...

A diferencia de métodos como *linear and nonlinear shooting method*<sup>1</sup>, entre otros, el método de diferencias finitas es bastante estable, en general (aunque requieran un poco más de poder computacional que los primeros).

Los métodos de diferencias finitas reemplazan cada una de las derivadas en la ecuación diferencial, con una aproximación de cociente de diferencias adecuada (esto no será tratado en detalle aquí, pero estas aproximaciones son básicamente distintos tipos de aproximaciones que emplean la expansión de Taylor de una función a diferentes órdenes para aproximar funciones y derivadas de distintos órdenes, dependiendo de la grilla que se elija o posea para llevar a cabo los cálculos).

El cociente de diferencias que se elija y el paso de los cálculos numéricos, dependen mucho del error que se desee.

Trabajando con funciones dependientes de 1 variable (este no será el caso a resolver, pues la ecuación de Poisson está definida a través de un problema multivariable), la primera y segunda derivada se aproximan de la siguiente manera:

$$y'(x_i) = \frac{1}{2h}[y(x_{i+1}) - y(x_{i-1})] - \frac{h^2}{6}y'''(\eta_i)$$
$$y''(x_i) = \frac{1}{h^2}[y(x_{i+1}) - 2y(x_i) + y(x_{i-1}))] - \frac{h^2}{12}y^{(4)}(\zeta_i)$$

Las variables independientes en la segunda y cuarta derivada no tienen mucha importancia, pues lo importante ahí es notar que ambos términos están asociados al error. Luego, se toman estas aproximaciones, se insertan en la ED, y luego se hace el cambio:

$$y(x_i) \rightarrow w_i$$

pudiendo con esto establecer un sistema de ecuaciones lineales (es decir, para resolverlo basta con llevar a cabo la inversión de una matriz).

### 2.2. Multivariable...

La ecuación diferencial en derivadas parciales que se desea resolver es del tipo Elíptica. La ecuación de Poisson tendrá aquí la siguiente forma:

$$\nabla^2 u(x, y) = \frac{\partial^2 u}{\partial x^2}(x, y) + \frac{\partial^2 u}{\partial y^2}(x, y) = f(x, y)$$

---

<sup>1</sup>R. Burden, J. Faires. Numerical Analysis. 9th edition. 2011.

la cual estará definida sobre una región  $R = \{(x, y) \mid a < x < b, c < y < d\}$ , con  $u(x, y) = g(x, y)$  para  $(x, y) \in S$ , con  $S$  denotando la frontera de  $R$ . Si  $f$  y  $g$  son continuas en sus dominios, entonces hay una solución única para esta ED.

Definido el problema de esta forma, y al ser 2 dimensional (muy útil este carácter de  $n = 2$  en estudios de conductividad sobre superficies, por ejemplo), se establecen dos pasos:  $h = (b - a)/n$  y  $k = (d - c)/m$ , con  $h$  y  $n$  correspondiendo con la derivada en  $x$ , y  $k$  y  $m$  con la derivada en  $y$ .

Se genera así la grilla  $(x_i, y_j)$  (más específicamente, estos puntos son los *puntos de malla* de la grilla, pero se les llama así a aquellos para los cuales  $i = 1, 2, \dots, n - 1$  y  $j = 1, 2, \dots, m - 1$ , con  $i$  y  $j$  por definirse a continuación), tales que:

$$\begin{aligned} x_i &= a + ih, \quad i \text{ hasta } n, \text{ desde } 0 \\ y_j &= c + jk, \quad j \text{ hasta } m, \text{ desde } 0 \end{aligned}$$

Se procede ahora con las expansiones a través de la serie de Taylor, generando así las fórmulas en diferencias centradas ( $i = 1, 2, \dots, n - 1$  y  $j = 1, 2, \dots, m - 1$ ):

$$\begin{aligned} \frac{\partial^2 u}{\partial x^2}(x_i, y_j) &= \frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j))}{h^2} - \frac{h^2}{12} \frac{\partial^4 u}{\partial x^4}(\zeta_i, y_j), \text{ con } \zeta_i \in (x_{i-1}, x_{i+1}) \\ \frac{\partial^2 u}{\partial y^2}(x_i, y_j) &= \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1}))}{k^2} - \frac{k^2}{12} \frac{\partial^4 u}{\partial y^4}(\eta_j, y_j), \text{ con } \eta_j \in (y_{j-1}, y_{j+1}) \end{aligned}$$

Así, la ED se vuelve bastante simple, desde un punto de vista numérico:

$$\begin{aligned} \frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j))}{h^2} + \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1}))}{k^2} = \\ f(x_i, y_j) + \frac{h^2}{12} \frac{\partial^4 u}{\partial x^4}(\zeta_i, y_j) + \frac{k^2}{12} \frac{\partial^4 u}{\partial y^4}(\eta_j, y_j) \end{aligned}$$

Todo esto resulta en el siguiente resumen para el método de diferencias finitas, aplicado a la ecuación de Poisson (haciendo el cambio  $u(x_i, y_j) \rightarrow w_{ij}$ ):

$$2[(h/k)^2 + 1]w_{ij} - (w_{i+1,j} + w_{i-1,j}) - (h/k)^2(w_{i,j+1} + w_{i,j-1}) = -h^2 f(x_i, y_j), \text{ para } i = 1, 2, \dots, n - 1 \text{ y } j = 1, 2, \dots, m - 1$$

y con condiciones de frontera:

$$\begin{aligned} w_{0j} &= g(x_0, y_j) \text{ y } w_{nj} = g(x_n, y_j), \text{ para cada } j = 0, 1, \dots, m \\ w_{i0} &= g(x_i, y_0) \text{ y } w_{im} = g(x_i, y_m), \text{ para cada } i = 1, \dots, n - 1 \end{aligned}$$

Esto produce un sistema de ecuaciones lineal siendo los puntos  $w_{ij}$  (los puntos interiores de la grilla) las incógnitas.

El siguiente es un algoritmo que permite solucionar el sistema de ecuaciones que se presenta (este algoritmo hace uso del método iterativo de Gauss-Seidel, por lo que permite que  $m \neq n$ ):

3. Sobre el empaquetado de código en C++ para creación de módulos para Python:
4. Sobre VPython:
5. Procedimientos generales: