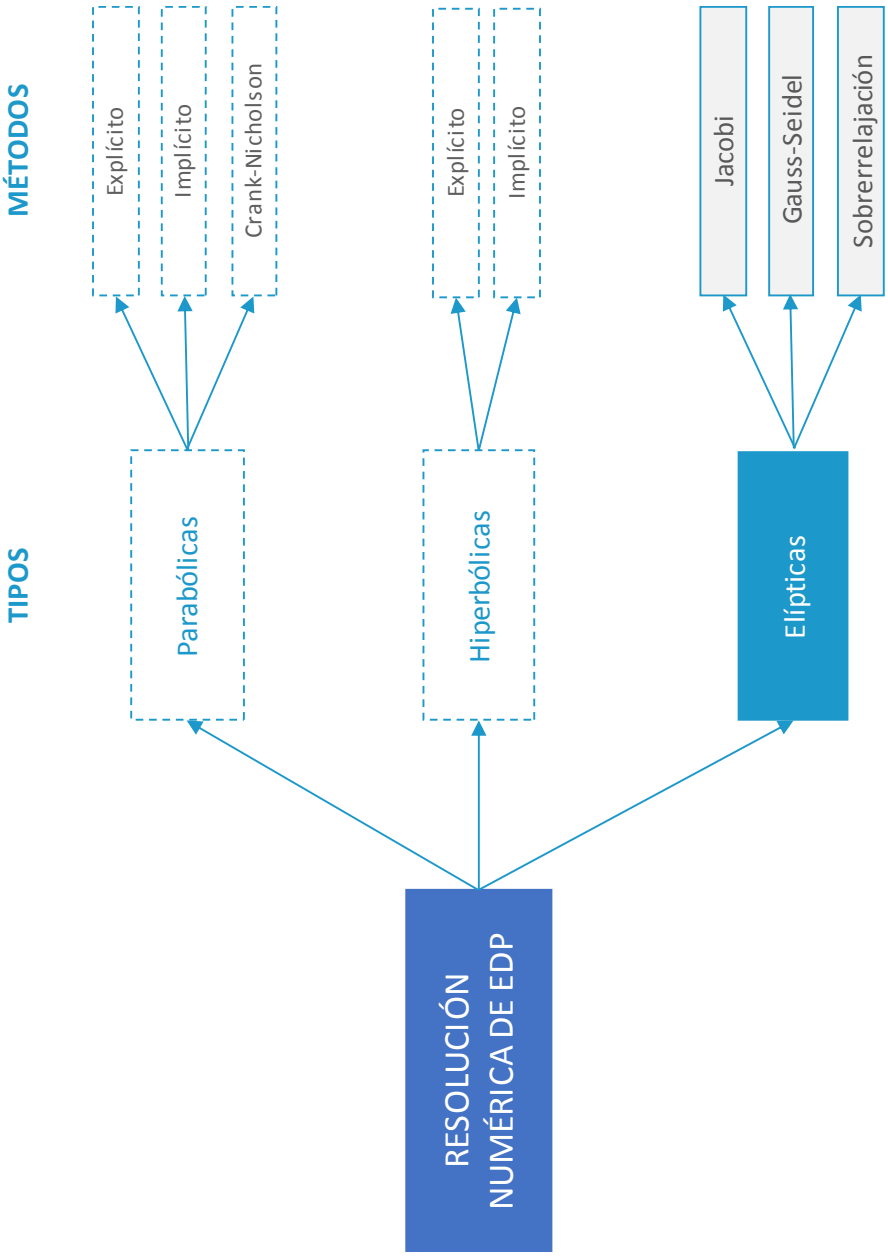
Métodos Numéricos Avanzados en Ingeniería

Problemas de contorno multidimensional. EDP elípticas

Índice

Esquema



Ideas clave

9.1. ¿Cómo estudiar este tema?

Para estudiar este tema lee las Ideas clave que encontrarás a continuación

Finalizamos el bloque de temas relacionados con las ecuaciones en derivadas parciales con aquellas que se caracterizan como elípticas. Dada la expresión general:

$$Au_{xx} + Bu_{xy} + Cu_{yy} + Du_x + Eu_y + Fu + G = 0$$

Las EDP elípticas cumplen $\Delta = B^2 - AC < 0$. Para poder resolver este tipo de EDP volveremos a utilizar la técnica de las diferencias finitas para transformar la EDP en una ecuación en diferencias. A partir de esta discretización, seremos capaces de obtener la solución aproximada en cada uno de los nodos.

Para estudiar este tema será necesario que sigas el desarrollo de los procedimientos de diferencias finitas para los diferentes métodos, y comprendas a la perfección **en qué nodos conocemos la información y en qué nodos tenemos que calcular las soluciones** numéricas.

Las ecuaciones de este tipo surgen de manera natural en el estudio de problemas físicos independientes del tiempo, como la distribución de calor en una región plana, la energía potencial de un punto en el plano bajo la acción de fuerzas gravitacionales y problemas estacionarios acerca de fluidos incompresibles. La descripción más sencilla de un problema de este tipo es:

$$\nabla^2 u(x,y) = 0 \leftrightarrow u_{xx} + u_{yy} = f(x,y), (x,y) \in R$$

siendo $R = \{(x,y): a < x < b, c < y < d\}$, sujeto a las condiciones de contorno:

$$u(x,y) = g(x,y), (x,y) \in S$$

siendo S la frontera de R. Bajo estas condiciones, obtenemos la discretización de la Figura 1.

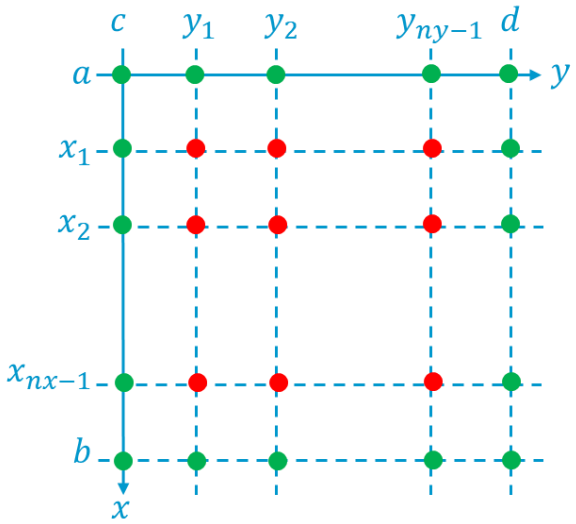


Figura 1. Información disponible en la EDP elíptica.

Los apartados de los que consta este tema son:

- Discretización de la EDP elíptica.
- Métodos iterativos para EDP elípticas:
  - Jacobi.
  - Gauss-Seidel.
  - Sobrerrelajación.
- Ejemplos resueltos de EDP elípticas.

9.2. Discretización de la EDP elíptica

Comencemos por plantear la transformación de la EDP en un esquema de diferencias finitas para, a continuación, plantear cómo resolver el sistema resultante.

Para realizar la transformación en una ecuación en diferencias vamos a aplicar diferencias centrales sobre u_{xx} y u_{tt}:

$$\frac{u(x+h,t)-2u(x,t)+u(x-h,t)}{h^2}+\frac{u(x,t+k)-2u(x,t)+u(x,t-k)}{k^2}$$
$$=f(x,y)\leftrightarrow \frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{h^2}+\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{k^2}=f_{i,j}$$
$$\leftrightarrow -2(\lambda^2+1)u_{i,j}+\left(u_{i+1,j}+u_{i-1,j}\right)+\lambda^2\left(u_{i,j+1}+u_{i,j-1}\right)=h^2f_{i,j}$$

Siendo $\lambda=h/k$. Si tomamos $h=k$ y $nx=ny=4$

$$4u_{i,j}-u_{i+1,j}-u_{i-1,j}-u_{i,j+1}-u_{i,j-1}=-h^2f_{i,j}, i=1,2,3, j=1,2,3.$$

Planteemos la ecuación en los diferentes nodos.

Para $j=1$:

$$4u_{i,1}-u_{i+1,1}-u_{i-1,1}-u_{i,2}-u_{i,0}=-h^2f_{i,1}\rightarrow$$
$$\rightarrow\begin{cases}4u_{1,1}-u_{2,1}-u_{0,1}-u_{1,2}-u_{1,0}=-h^2f_{1,1}\\4u_{2,1}-u_{3,1}-u_{1,1}-u_{2,2}-u_{2,0}=-h^2f_{2,1}\\4u_{3,1}-u_{4,1}-u_{2,1}-u_{3,2}-u_{3,0}=-h^2f_{3,1}\end{cases}$$

Para $j=2$:

$$4u_{i,2}-u_{i+1,2}-u_{i-1,2}-u_{i,3}-u_{i,1}=-h^2f_{i,2}\rightarrow$$
$$\rightarrow\begin{cases}4u_{1,2}-u_{2,2}-u_{0,2}-u_{1,3}-u_{1,1}=-h^2f_{1,2}\\4u_{2,2}-u_{3,2}-u_{1,2}-u_{2,3}-u_{2,1}=-h^2f_{2,2}\\4u_{3,2}-u_{4,2}-u_{2,2}-u_{3,3}-u_{3,1}=-h^2f_{3,2}\end{cases}$$

Para $j=3$:

$$4u_{i,3}-u_{i+1,3}-u_{i-1,3}-u_{i,4}-u_{i,2}=-h^2f_{i,3}\rightarrow$$

$$\rightarrow \begin{cases} 4u_{1,3} - u_{2,3} - u_{0,3} - u_{1,4} - u_{1,2} = -h^2 f_{1,3} \\ 4u_{2,3} - u_{3,3} - u_{1,3} - u_{2,4} - u_{2,2} = -h^2 f_{2,3} \\ 4u_{3,3} - u_{4,3} - u_{2,3} - u_{3,4} - u_{3,2} = -h^2 f_{3,3} \end{cases}$$

En este caso no vamos a plantear un sistema diferente para cada instante j, sino que vamos a resolverlo para todos los instantes j a la vez. Para ello, ordenamos las variables secuencialmente, como se ilustra en la Figura 2.

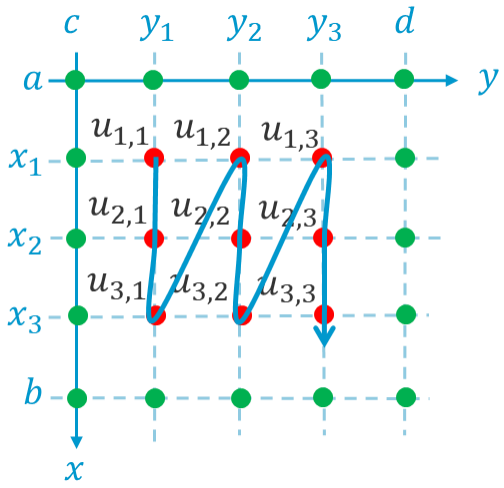


Figura 2. Organización de los nodos solución.

Esta organización da como resultado el siguiente sistema de ecuaciones lineales.

$$\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} u_{1,1} \\ u_{2,1} \\ u_{3,1} \\ u_{1,2} \\ u_{2,2} \\ u_{3,2} \\ u_{1,3} \\ u_{2,3} \\ u_{33} \end{bmatrix} = -h^2 \begin{bmatrix} f_{1,1} \\ f_{2,1} \\ f_{3,1} \\ f_{1,2} \\ f_{2,2} \\ f_{3,2} \\ f_{1,3} \\ f_{2,3} \\ f_{3,3} \end{bmatrix} + \begin{bmatrix} u_{0,1} + u_{1,0} \\ u_{2,0} \\ u_{3,0} + u_{4,1} \\ u_{0,2} \\ 0 \\ u_{4,2} \\ u_{0,3} + u_{1,4} \\ u_{2,4} \\ u_{4,3} + u_{3,4} \end{bmatrix}$$

Si nos resulta más sencillo, podemos aplicar el cambio de variables en los subíndices:

$$l = i + (ny - 1)(j - 1), i = 1,2, \dots, nx - 1, j = 1,2, \dots, ny - 1$$

que son los puntos donde queremos obtener la solución, cuya asignación se muestra en la Tabla 1.

i	j	l
1	1	1
2	1	2
3	1	3
1	2	4
2	2	5
3	2	6
1	3	7
2	3	8
3	3	9

Tabla 1. Asignación del índice l a partir del cambio de variables.

De este modo, renombrando los nodos de la solución como $P_{i,j} = P_l$ podríamos reescribir el problema como:

$$\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \\ P_7 \\ P_8 \\ P_9 \end{bmatrix} = -h^2 \begin{bmatrix} f_{1,1} \\ f_{2,1} \\ f_{3,1} \\ f_{1,2} \\ f_{2,2} \\ f_{3,2} \\ f_{1,3} \\ f_{2,3} \\ f_{3,3} \end{bmatrix} + \begin{bmatrix} u_{0,1} + u_{1,0} \\ u_{2,0} \\ u_{3,0} + u_{4,1} \\ u_{0,2} \\ 0 \\ u_{4,2} \\ u_{0,3} + u_{1,4} \\ u_{2,4} \\ u_{4,3} + u_{3,4} \end{bmatrix}$$

La solución pasa por resolver este sistema lineal. Podemos observar que se trata de un sistema de tamaño $(n-1)^3$, siendo este un valor muy grande. Asimismo, la matriz tiene una gran cantidad de ceros. De modo que el uso de métodos directos para resolver este tipo de sistemas queda totalmente desaconsejado por sus características de inestabilidad.

La alternativa pasa por el uso de métodos iterativos, para los cuales el sistema sí es estable.

9.3. Métodos iterativos para resolver EDP elípticas

Para resolver la EDP elíptica vamos a utilizar métodos iterativos, puesto que estos garantizan la estabilidad de la solución. Los métodos iterativos para resolver sistemas lineales los trataremos en el tema 10. Sin embargo, aquí daremos una breve introducción para resolver las EDP elípticas.

Partiremos de la discretización:

$$2(\lambda^2+1)u_{i,j} - \left(u_{i+1,j} + u_{i-1,j}\right) - \lambda^2\left(u_{i,j+1} + u_{i,j-1}\right) = -h^2 f_{i,j} \leftrightarrow$$
$$\leftrightarrow u_{i,j} = \frac{1}{2(\lambda^2+1)}\left[u_{i+1,j} + u_{i-1,j} + \lambda^2\left(u_{i,j+1} + u_{i,j-1}\right) - h^2 f_{i,j}\right]$$

resolviendo en los nodos $i = 1,2,\dots,nx-1, j = 1,2,\dots,ny-1$. Denotaremos por $u_{i,j}^{(iter)}$ la iteración $iter$. A partir de una estimación inicial $u^{(0)}$ obtendremos mediante un método iterativo la solución de la EDP elíptica.

El **¡Error! No se encuentra el origen de la referencia.** algoritmo muestra los pasos a seguir para resolver la EDP elíptica con el método de diferencias finitas a partir de métodos iterativos.

Resolución de la EDP elíptica:

$$u_{xx}(x,y) + u_{yy}(x,y) = f(x,y), (x,y) \in R = (a,b) \times (c,d)$$
$$u(x,y) = g(x,y), (x,y) \in S$$

ENTRADA: Valores a, b, c, d de la frontera; enteros nx, ny; funciones f, g; tolerancia TOL; número máximo de iteraciones $maxiter$.

SALIDA: Valores $x, y, u(x,y)$ o mensaje de fracaso.

1. Definición de los espaciados h, t y del parámetro λ

2. Definición de las variables discretizadas: $x_i = a + ih, i = 1, \dots, nx - 1, y_j = c + jk, j = 1,2, \dots, ny - 1$
3. Inicialización de la matriz $u(x, y)$ con ceros
4. Actualización de $u(x, y)$ con las condiciones de contorno $g(x, y)$
5. Inicializar variables de control del método iterativo: $iter = 1, incre = tol + 1, u^{(0)} = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix}$.
6. Mientras $iter < maxiter$ y $incre > tol$
   6.1. Utilizar un método iterativo para resolver el sistema lineal y obtener u.
   6.2. $incre = norm\left(u - u^{(0)}\right)$.
   6.3. $u^{(0)} = u$
   6.4. $iter = iter + 1$
7. Si se sale del bucle anterior porque se ha alcanzado la solución, sacar la solución. Si no, sacar mensaje de fracaso.
8. FIN.

A continuación, vamos a presentar una serie de métodos iterativos que nos van a permitir implementar el paso 6.1. del algoritmo que acabamos de ver y, por tanto, obtener la solución numérica de la EDP elíptica.

Método de Jacobi

La expresión iterativa del método de Jacobi es:

$$u_{i,j}^{(iter+1)} = \frac{1}{2(\lambda^2 + 1)}\left[u_{i+1,j}^{(iter)} + u_{i-1,j}^{(iter)} + \lambda^2\left(u_{i,j+1}^{(iter)} + u_{i,j-1}^{(iter)}\right) - h^2 f_{i,j}\right]$$

en el rango de índices $i = 1,2, \dots, nx - 1, j = 1,2, \dots, ny - 1$, tomando como criterio de parada:

$$\max_i \left\{ \max_j \left\{ \left| u_{i,j}^{(iter+1)} - u_{i,j}^{(iter)} \right| \right\} \right\} < tol$$

Por tanto, para calcular el elemento $u_{i,j}$ en la siguiente iteración es necesario conocer los valores de sus nodos contiguos en la iteración anterior, como ilustra la Figura 3.

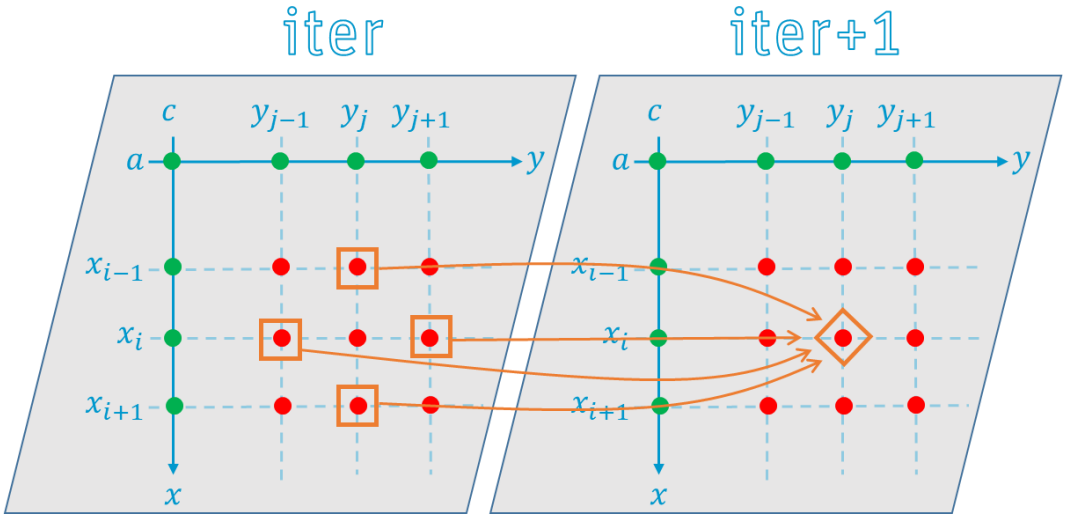


Figura 3. Estructura de nodos para la obtención de $u_{i,j}$ en el iterado actual con el método de Jacobi.

Para implementar el método, sería necesario utilizar un bucle para cada una de las variables. El pseudocódigo que se expone a continuación muestra una posible implementación del método de Jacobi en Matlab.

```
Para i=2, …, nx
    Para j=2, …, ny
        U(i,j)=1/(2*(lambda^2+1))*(U0(i+1,j)+U0(i-1,j)+…
        lambda^2*(U0(i,j+1)+U0(i,j-1)-h^2*f(i,j)));
    Fin para j
Fin para i
```

Resolvamos, con el método de Jacobi, el

Ejemplo 1. Sea la EDP:

$$u_{xx} + u_{yy} = 0, (x, y) \in R = \{(x, y): 0 < x < 1, 0 < y < 1\}$$

con las condiciones de contorno dadas por:

$$u(x, 0) = u(y, 0) = 0, u(x, 1) = 100x, u(1, y) = 100y$$

Tomando $h = k = 0.25$, obtén la solución por el método de Jacobi hasta que los valores varíen menos de 0.01.

Planteamos el esquema en diferencias finitas. Si utilizamos los desarrollos anteriores, podemos observar que $f(x, y) = 0$, de modo que el sistema que tenemos que resolver es:

$$u_{i,j} = \frac{1}{2(\lambda^2 + 1)}\left[u_{i+1,j} + u_{i-1,j} + \lambda^2\left(u_{i,j+1} + u_{i,j-1}\right)\right]$$

en los nodos $i = 1,2,3, j = 1,2,3$.

Introducimos los valores en Matlab:

```
>> a=0; b=1; nx=1/.25; c=0; d=1; nt=1/.25;
>> f=@(x,y) 0*x*y;
>> uxc=@(x) 0*x; uxd=@(x) 100*x;
>> uya=@(y) 0*y; uyb=@(y) 100*y;
>> tol=1e-2; maxiter=100;
```

Y ejecutamos el programa que hayamos creado para utilizar el método de Jacobi. El método ha tardado 24 iteraciones. Los resultados son:

	$y = 0$	$y = 0.25$	$y = 0.5$	$y = 0.75$	$y = 1$
$x = 0$	0	0	0	0	0
$x = 0.25$	0	6.2408	12.4878	18.7408	25
$x = 0.5$	0	12.4878	24.9817	37.4878	50
$x = 0.75$	0	18.7408	37.4878	56.2408	75
$x = 1$	0	25	50	75	100

Ejemplo 1. Sea la EDP:

$$u_{xx} + u_{yy} = 0, (x, y) \in R = \{(x, y): 0 < x < 1, 0 < y < 1\}$$

con las condiciones de contorno dadas por:

$$u(x, 0) = u(y, 0) = 0, u(x, 1) = 100x, u(1, y) = 100y$$

Tomando $h = k = 0.25$, obtén la solución por el método de Jacobi hasta que los valores varíen menos de 0.01.

Planteamos el esquema en diferencias finitas. Si utilizamos los desarrollos anteriores, podemos observar que $f(x, y) = 0$, de modo que el sistema que tenemos que resolver es:

$$u_{i,j} = \frac{1}{2(\lambda^2 + 1)}\left[u_{i+1,j} + u_{i-1,j} + \lambda^2\left(u_{i,j+1} + u_{i,j-1}\right)\right]$$

en los nodos $i = 1,2,3, j = 1,2,3$.

Introducimos los valores en Matlab:

```
>> a=0; b=1; nx=1/.25; c=0; d=1; nt=1/.25;
>> f=@(x,y) 0*x*y;
>> uxc=@(x) 0*x; uxd=@(x) 100*x;
>> uya=@(y) 0*y; uyb=@(y) 100*y;
>> tol=1e-2; maxiter=100;
```

Y ejecutamos el programa que hayamos creado para utilizar el método de Jacobi. El método ha tardado 24 iteraciones. Los resultados son:

	$y = 0$	$y = 0.25$	$y = 0.5$	$y = 0.75$	$y = 1$
$x = 0$	0	0	0	0	0
$x = 0.25$	0	6.2408	12.4878	18.7408	25
$x = 0.5$	0	12.4878	24.9817	37.4878	50
$x = 0.75$	0	18.7408	37.4878	56.2408	75
$x = 1$	0	25	50	75	100

Tabla 2. Resultados de EDP por el método Jacobi.

Método de Gauss-Seidel

El método de Gauss-Seidel se basa en la idea de que los valores que vamos obteniendo en una iteración deben estar **más próximos a la solución que los de la iteración anterior,** por lo que pueden usarse a medida que se van obteniendo para calcular el resto de valores. A diferencia del método de Jacobi, **el orden en que se actualizan los valores influye** en el resultado.

La expresión iterativa del método de Gauss-Seidel es:

$$u_{i,j}^{(iter+1)} = \frac{1}{2(\lambda^2+1)}\left[u_{i+1,j}^{(iter)} + u_{i-1,j}^{(iter+1)} + \lambda^2\left(u_{i,j+1}^{(iter)} + u_{i,j-1}^{(iter+1)}\right) - h^2 f_{i,j}\right]$$

en el rango de índices $i = 1,2,\dots,nx-1, j = 1,2,\dots,ny-1$.

Cabe esperar una convergencia más rápida que en el método de Jacobi. Además, no necesitamos guardar los valores de la iteración anterior, que se pueden sobrescribir con los que se van calculando en la iteración actual.

Por tanto, para conocer el valor del elemento $u_{i,j}$ del iterado actual tendremos en cuenta sus valores contiguos. En este caso, si ya están calculados, utilizaremos los de la iteración actual; si no están calculados, utilizaremos los de la iteración anterior. La Figura 4 ilustra esta relación.

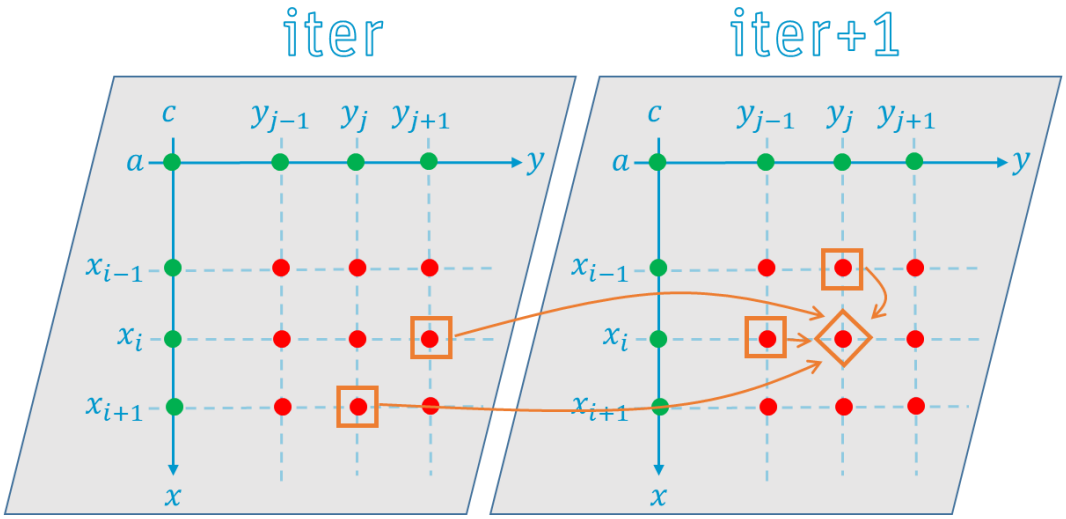


Figura 4. Estructura de nodos para la obtención de $u_{i,j}$ en el iterado actual con el método de Gauss-Seidel.

Para implementar el método, sería necesario utilizar un bucle para cada una de las variables. El siguiente pseudocódigo muestra una posible implementación del método de Gauss-Seidel en Matlab.

```
Para i=2, …, nx
    Para j=2, …, ny
        U(i,j)=1/(2*(lambda^2+1))*(U0(i+1,j)+U(i-1,j)+…
        lambda^2*(U0(i,j+1)+U(i,j-1)-h^2*f(i,j)));
    Fin para j
Fin para i
```

Resolvamos, con el método de Gauss-Seidel, el Ejemplo 2.

> Ejemplo 2. Sea la EDP:
>
> $$u_{xx} + u_{yy} = 0, (x, y) \in R = \{(x, y): 0 < x < 1, 0 < y < 1\}$$
>
> con las condiciones de contorno dadas por:
>
> $$u(x, 0) = u(y, 0) = 0, u(x, 1) = 100x, u(1, y) = 100y$$
>
> Tomando $h = k = 0.25$, obtén la solución por el método de Gauss-Seidel hasta que los valores varíen menos de 0.01.

Planteamos el esquema en diferencias finitas. Si utilizamos los desarrollos anteriores, podemos observar que $f(x,y)=0$, de modo que el sistema que tenemos que resolver es:

$$u_{i,j}=\frac{1}{2(\lambda^2+1)}\left[u_{i+1,j}+u_{i-1,j}+\lambda^2\left(u_{i,j+1}+u_{i,j-1}\right)\right]$$

en los nodos $i=1,2,3, j=1,2,3$.

Introducimos los valores en Matlab:

```
>> a=0; b=1; nx=1/.25; c=0; d=1; nt=1/.25;
>> f=@(x,y) 0*x*y;
>> uxc=@(x) 0*x; uxd=@(x) 100*x;
>> uya=@(y) 0*y; uyb=@(y) 100*y;
>> tol=1e-2; maxiter=100;
```

Y ejecutamos el programa que hayamos creado para utilizar el método de Gauss-Seidel. El método ha tardado 15 iteraciones. Los resultados son:

	$y=0$	$y=0.25$	$y=0.5$	$y=0.75$	$y=1$
$x=0$	0	0	0	0	0
$x=0.25$	0	6.2416	12.4916	18.7458	25
$x=0.5$	0	12.4916	24.9916	37.4958	50
$x=0.75$	0	18.7458	37.4958	56.2479	75
$x=1$	0	25	50	75	100

Tabla 3. Resultados EDP 2. Método de Gauss-Seidel.

Método de sobrerrelajación (SOR)

Si en cada una de las iteraciones de Gauss-Seidel nos estamos acercando a la solución, podemos potenciar este acercamiento multiplicando el desplazamiento realizado por un factor ω. Este factor toma el nombre de factor de relajación, y debe estar contenido en el intervalo $[0,2]$. En función del valor de ω tendremos un método convergente o no convergente.

La idea pasa por la ejecución del método de Gauss-Seidel en cada uno de los nodos y, a continuación, realizar el ajuste con el factor ω. Así, la expresión iterativa del método SOR es:

$$\hat{u}_{i,j}^{(iter+1)} = \frac{1}{2(\lambda^2+1)}\left[u_{i+1,j}^{(iter)} + u_{i-1,j}^{(iter+1)} + \lambda^2\left(u_{i,j+1}^{(iter)} + u_{i,j-1}^{(iter+1)}\right) - h^2 f_{i,j}\right]$$

$$u_{i,j}^{(iter+1)} = (1-\omega)u_{i,j}^{(iter)} + \omega\hat{u}_{i,j}^{(iter+1)}$$

en el rango de índices $i = 1,2,\ldots,nx-1, j = 1,2,\ldots,ny-1$.

El método de Gauss-Seidel es un caso particular del método SOR, en el que $\omega = 1$. Es posible encontrar un valor óptimo de ω, aunque en la práctica se determina de forma heurística.

Para implementar el método, modificaremos el pseudocódigo anterior de modo que incluyamos la segunda línea del método. El **¡Error! No se encuentra el origen de la referencia.**muestra una posible implementación del método SOR en Matlab.

```
Para i=2, …, nx
      Para j=2, …, ny
            UGS(i,j)=1/(2*(lambda^2+1))*(U0(i+1,j)+U(i-1,j)+…
            lambda^2*(U0(i,j+1)+U(i,j-1)-h^2*f(i,j)));
            U(i,j)=(1-w)*U0(i,j)+w*UGS(i,j);
```

Fin para `j`

Fin para `i`

Resolvamos, con el método SOR, el Ejemplo 3:

Ejemplo 3. Sea la EDP:

$$u_{xx} + u_{yy} = 0, (x, y) \in R = \{(x, y): 0 < x < 1, 0 < y < 1\}$$

con las condiciones de contorno dadas por:

$$u(x, 0) = u(y, 0) = 0, u(x, 1) = 100x, u(1, y) = 100y$$

Tomando $h = k = 0.25$, obtén la solución por el método SOR con $\omega = 1.2$ hasta que los valores varíen menos de 0.01.

Planteamos el esquema en diferencias finitas. Si utilizamos los desarrollos anteriores, podemos observar que $f(x, y) = 0$, de modo que el sistema que tenemos que resolver es:

$$u_{i,j} = \frac{1}{2(\lambda^2 + 1)}\left[u_{i+1,j} + u_{i-1,j} + \lambda^2\left(u_{i,j+1} + u_{i,j-1}\right)\right]$$

en los nodos $i = 1,2,3, j = 1,2,3$.

Introducimos los valores en Matlab:

```
>> a=0; b=1; nx=1/.25; c=0; d=1; nt=1/.25;
>> f=@(x,y) 0*x*y;
>> uxc=@(x) 0*x; uxd=@(x) 100*x;
>> uya=@(y) 0*y; uyb=@(y) 100*y;
>> tol=1e-2; maxiter=100;
```

Y ejecutamos el programa que hayamos creado para utilizar el método SOR. El método ha tardado 10 iteraciones. Los resultados son:

	$y = 0$	$y = 0.25$	$y = 0.5$	$y = 0.75$	$y = 1$
$x = 0$	0	0	0	0	0
$x = 0.25$	0	6.2526	12.5014	18.7503	25
$x = 0.5$	0	12.5014	25.0006	37.5003	50

$x = 0.75$	0	18.7503	37.5003	56.2500	75
$x = 1$	0	25	50	75	100

Tabla 4. Resultados de EDP 3. Método SOR.

9.4. Ejemplos resueltos de EDP elípticas

En esta sección vamos a plantear una serie de EDP elípticas y sus resultados numéricos. En el **Ejemplo 4** veremos cómo obtener la ecuación en diferencias a partir de una EDP parecida a la que hemos desarrollado a lo largo del tema.

Ejemplo 4. Sea la EDP:

$$u_{xx} + u_{yy} = -(\cos(x+y) + \cos(x-y)), x \in (0,\pi), y \in \left(0,\frac{\pi}{2}\right)$$

$$u(0,y) = \cos(y), u(\pi,y) = -\cos(y), y \in \left[0,\frac{\pi}{2}\right]$$

$$u(x,0) = \cos(x), u\left(x,\frac{\pi}{2}\right) = 0, x \in [0,\pi]$$

Tomando $h = \pi/10$ y $k = \pi/20$, y utilizando una tolerancia de 10^{-10}, obtén la solución $u\left(\frac{3\pi}{10}, y\right)$ a partir de los métodos de Jacobi, Gauss-Seidel y SOR ($\omega = 1.2$), indicando el número de iteraciones.

Si hemos implementado los programas correctamente, este ejercicio tiene solo la complicación de introducir los datos en la consola de Matlab.

```
>> a=0; b=pi; nx=10;
>> c=0; d=pi/2; ny=10;
>> tol=1e-10;
>> uay=@(y) cos(y); uby=@(y) -cos(y);
>> ucx=@(x) cos(x); udx=@(x) 0*x;
>> f=@(x,y) -(cos(x+y)+cos(x-y));
```

Los resultados son los siguientes:

	Jacobi	Gauss-Seidel	SOR
Iteraciones	259	192	131
$y = 0$	0.5878	0.5878	0.5878
$y = \frac{\pi}{20}$	0.7931	0.7931	0.7931
$y = \frac{\pi}{10}$	0.9100	0.9100	0.9100
$y = \frac{3\pi}{20}$	0.9511	0.9511	0.9511
$y = \frac{\pi}{5}$	0.9280	0.9280	0.9280
$y = \frac{5\pi}{20}$	0.8517	0.8517	0.8517
$y = \frac{3\pi}{10}$	0.7320	0.7320	0.7320
$y = \frac{7\pi}{20}$	0.5783	0.5783	0.5783
$y = \frac{2\pi}{5}$	0.3994	0.3994	0.3994
$y = \frac{9\pi}{20}$	0.2038	0.2038	0.2038
$y = \frac{\pi}{2}$	0	0	0

Tabla 5. Resultados EDP 4. Métodos Jacobi, Gauss Seide y SOR.

En el Ejemplo 5 veremos un caso en que las condiciones están en función de las derivadas.

Ejemplo 5. Sea la EDP:

$$u_{xx} + u_{yy} = 0, x \in [-1,1], y \in [-1,1]$$
$$u(x,-1) = 1, u(x,1) = 0, x \in [-1,1]$$
$$u_x(-1,y) = -\frac{1}{2}u(-1,y), u_x(1,y) = u(1,y), \qquad y \in [-1,1]$$

Considerando $nx = ny = 8$, una tolerancia de 10^{-5} y diferencias finitas de segundo orden en la aproximación de las derivadas, responde los siguientes apartados.

a) Obtén la solución del problema con el método de Jacobi, indicando la solución en $y \in \{-0.5,0,0.5\}$ y el número de iteraciones.
b) Obtén la solución del problema con el método de Gauss-Seidel, indicando la solución en $y \in \{-0.5,0,0.5\}$ y el número de iteraciones.

c) Obtén la solución del problema con el método SOR para $\omega = 0.1, 0.2, \dots, 1.9$. Representa en una gráfica la relación entre el número de iteraciones y el valor de ω.

En primer lugar, vamos a discretizar el problema. La expresión general la hemos visto a lo largo del tema, y es:

$$u_{i,j} = \frac{1}{2(\lambda^2+1)}\left[u_{i+1,j} + u_{i-1,j} + \lambda^2\left(u_{i,j+1} + u_{i,j-1}\right)\right]$$

En los nodos $i = 0,1,2,\dots,8, j = 0,1,2,\dots,8$. Vayamos a las condiciones de contorno:

$$u_x(x,y) \approx \frac{u(x+h,y) - u(x-h.y)}{2h}$$

$$\rightarrow \begin{cases} u_x(-1,y) \approx \dfrac{u(-1+h,y) - u(-1-h,y)}{2h} \\ u_x(1,y) \approx \dfrac{u(1+h,y) - u(1-h,y)}{2h} \end{cases}$$

Tomando las condiciones que nos aporta el enunciado:

$$u_x(-1,y) = \frac{1}{2}u(-1,y) \leftrightarrow \frac{u(-1+h,y) - u(-1-h,y)}{2h}$$
$$= -\frac{1}{2}u(-1,y) \rightarrow \frac{u_{1,j} - u_{-1,j}}{2h} = -\frac{1}{2}u_{0,j}$$
$$\leftrightarrow u_{-1,j} = u_{1,j} + hu_{0,j}$$
$$u_x(1,y) = u(1,y) \leftrightarrow \frac{u(1+h,y) - u(1-h,y)}{2h} = u(1,y)$$
$$\rightarrow \frac{u_{nx+1,j} - u_{nx-1,j}}{2h} = u_{nx,j} \leftrightarrow u_{nx+1}$$
$$= 2hu_{nx,j} + u_{nx-1,j}$$

Estas dos condiciones de contorno habrá que evaluarlas para $i = 0$ y para $i = nx$. Por tanto, en el nodo $i = 0$ habrá que obtener:

$$u_{0,j} = \frac{1}{2(\lambda^2+1)}\left[u_{1,j} + u_{-1,j} + \lambda^2\left(u_{0,j+1} + u_{0,j-1}\right)\right]$$
$$= \frac{1}{2(\lambda^2+1)}\left[u_{1,j} + u_{1,j} + hu_{0,j} + \lambda^2\left(u_{0,j+1} + u_{0,j-1}\right)\right]$$
$$= \frac{1}{2(\lambda^2+1)}\left[2u_{1,j} + hu_{0,j} + \lambda^2\left(u_{0,j+1} + u_{0,j-1}\right)\right] \leftrightarrow$$
$$\leftrightarrow u_{0,j} = \frac{1}{2(\lambda^2+1)-h}\left[2u_{1,j} + \lambda^2\left(u_{0,j+1} + u_{0,j-1}\right)\right]$$

En el nodo $i = nx$, resolveremos:

$$u_{nx,j} = \frac{1}{2(\lambda^2+1)}\left[u_{nx+1,j} + u_{nx-1,j} + \lambda^2\left(u_{nx,j+1} + u_{nx,j-1}\right)\right]$$
$$= \frac{1}{2(\lambda^2+1)}\left[2hu_{nx,j} + u_{nx-1,j} + u_{nx-1,j} + \lambda^2\left(u_{nx,j+1} + u_{nx,j-1}\right)\right]$$
$$= \frac{1}{2(\lambda^2+1)}\left[2hu_{nx,j} + 2u_{nx-1,j} + \lambda^2\left(u_{nx,j+1} + u_{nx,j-1}\right)\right] \leftrightarrow$$
$$\leftrightarrow u_{nx,j} = \frac{1}{2(\lambda^2+1-h)}\left[2u_{nx-1,j} + \lambda^2\left(u_{nx,j+1} + u_{nx,j-1}\right)\right]$$

a) Al aplicar el método de Jacobi, las expresiones quedan como:

$$u_{0,j}^{(iter+1)} = \frac{1}{2(\lambda^2+1)-h}\left[2u_{1,j}^{(iter)} + \lambda^2\left(u_{0,j+1}^{(iter)} + u_{0,j-1}^{(iter)}\right)\right], j = 2, \dots, ny-1$$

$$u_{i,j}^{(iter+1)} = \frac{1}{2(\lambda^2+1)}\left[u_{i+1,j}^{(iter)} + u_{i-1,j}^{(iter)} + \lambda^2\left(u_{i,j+1}^{(iter)} + u_{i,j-1}^{(iter)}\right)\right], i = 1, \dots, nx, j = 2, \dots, ny-1$$

$$u_{nx,j}^{(iter+1)} = \frac{1}{2(\lambda^2+1-h)}\left[2u_{nx-1,j}^{(iter)} + \lambda^2\left(u_{nx,j+1}^{(iter)} + u_{nx,j-1}^{(iter)}\right)\right], j = 2, \dots, ny-1$$

Al ejecutar el método de Jacobi, se converge en 401 iteraciones, obteniendo los siguientes resultados:

x	$u(x,-0.5)$	$u(x,0)$	$u(x,0.5)$
-1	1.1219	0.9212	0.5171
-0.75	1.0190	0.8316	0.4668
-0.5	0.9698	0.7869	0.4425
-0.25	0.9610	0.7838	0.4432
0	0.9890	0.8228	0.4701
0.25	1.0580	0.9101	0.5273
0.5	1.1817	1.0578	0.6214
0.75	1.3867	1.2857	0.7625
1	1.7189	1.6207	0.9646

Tabla 6. Resultados EDP 5. Método Jacobi.

b) Para aplicar el método de Gauss-Seidel, tenemos que conocer en cada instante los puntos que ya hemos calculado. Las expresiones quedan como:

$$u_{0,j}^{(iter+1)} = \frac{1}{2(\lambda^2+1)-h}\Big[2u_{1,j}^{(iter)} + \lambda^2\Big(u_{0,j+1}^{(iter)} + u_{0,j-1}^{(iter+1)}\Big)\Big], j = 2,\dots,ny-1$$

$$u_{i,j}^{(iter+1)} = \frac{1}{2(\lambda^2+1)}\Big[u_{i+1,j}^{(iter)} + u_{i-1,j}^{(iter+1)} + \lambda^2\Big(u_{i,j+1}^{(iter)} + u_{i,j-1}^{(iter+1)}\Big)\Big], i = 1,\dots,nx, j = 2,\dots,ny-1$$

$$u_{nx,j}^{(iter+1)} = \frac{1}{2(\lambda^2+1-h)}\Big[2u_{nx-1,j}^{(iter+1)} + \lambda^2\Big(u_{nx,j+1}^{(iter)} + u_{nx,j-1}^{(iter+1)}\Big)\Big], j = 2,\dots,ny-1$$

Al ejecutar el método de Gauss-Seidel, se converge en 215 iteraciones, obteniendo los siguientes resultados:

x	$u(x,-0.5)$	$u(x,0)$	$u(x,0.5)$
-1	1.1219	0.9213	0.5171
-0.75	1.0191	0.8317	0.4668
-0.5	0.9698	0.7870	0.4425
-0.25	0.9611	0.7839	0.4432
0	0.9890	0.8229	0.4702
0.25	1.0580	0.9102	0.5274
0.5	1.1818	1.0580	0.6215
0.75	1.3868	1.2859	0.7627
1	1.7191	1.6209	0.9648

Tabla 7. Resultados EDP 5. Método Gauss-Seidel.

c) La aplicación del método SOR se basa en el método de Gauss-Seidel, pero añadiendo un último paso en que:

$$u_{i,j}^{(iter+1)} = (1-\omega)u_{i,j}^{(iter)} + \omega\hat{u}_{i,j}^{(iter+1)}$$

donde $\hat{u}_{i,j}^{(iter+1)}$ es el resultado obtenido con Gauss-Seidel. Ejecutamos el programa para diferentes valores de ω como indica el enunciado y obtenemos los siguientes resultados:

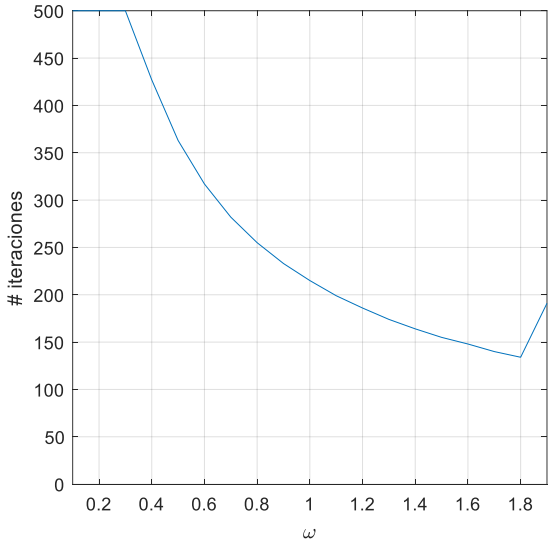


Figura 5. Representación de resultados de EDP 5. Método SOR.

Lo + recomendado

Lecciones magistrales

EDP elíptica en coordenadas polares

En ocasiones, la geometría del problema nos invita a utilizar unas coordenadas que no sean cartesianas. En el problema que vamos a resolver en esta lección magistral, se va a dar ese caso. El problema que vamos a resolver es:

$$u_{xx} + u_{yy} = 0, x^2 + y^2 < 1, y > 0$$

con las condiciones de contorno:

$$u(x,0) = 0, x \in (-1,1); u(x,y) = 100, x^2 + y^2 = 1, y > 0$$

Accede a la lección magistral a través del aula virtual

No dejes de leer

Diferencias finitas para problemas elípticos

Blanes, S., Ginestar, D. y Roselló, M. D. (2014). «Diferencias finitas para problemas elípticos». En Autores, *Introducción a los métodos numéricos para ecuaciones diferenciales* (2ª Ed.). Valencia: Editorial de la Universidad Politécnica de Valencia.

En el capítulo 5.4 de este libro se plantean las EDP elípticas. Para ello, se desarrolla la solución aplicando diferencias finitas sobre la EDP conocida como la **ecuación de Poisson.**

Accede al libro a través de la Biblioteca Virtual de UNIR

Diferencias finitas: ecuaciones elípticas

Chapra, S. C. y Canale, R. P. (2007). «Diferencias finitas: ecuaciones elípticas». En Autores, *Métodos numéricos para ingenieros* (5ª Ed.). Madrid: McGraw-Hill.

En el capítulo 29 de este libro se plantea el método de diferencias finitas para resolver EDP elípticas. A partir de la **ecuación de Laplace**, se desarrolla el método de diferencias finitas para obtener el sistema resultante. Para la resolución **a partir de métodos iterativos**, se muestran los métodos de Gauss-Seidel y sobrerrelajación.

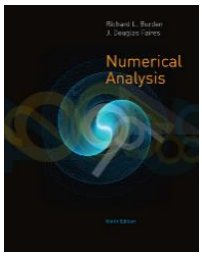
Accede al libro a través de la Biblioteca Virtual de UNIR

+ Información

A fondo

Ecuaciones en derivadas parciales elípticas

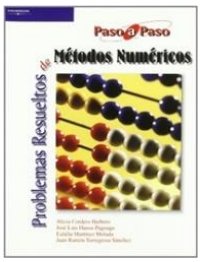
Burden, R. L. y Faires, J. D. (2011). «Ecuaciones en derivadas parciales elípticas». En Autores, *Numerical analysis (9ª Ed)*. Boston: Brooks/Cole CENGAGE learning.



En el Capítulo 12.1 de *Numerical Analysis* se desarrolla el problema de las EDP elípticas, desarrollando la discretización realizada y presentando **alternativas iterativas para la resolución del sistema lineal**. Además, en la parte final de la sección, se plantean una serie de ejercicios.

Ecuaciones elípticas

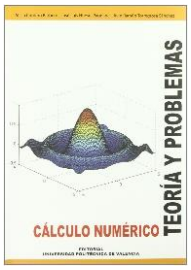
Cordero, A., Hueso, J. L., Martínez, E. y Torregrosa, J. R. (2006). «Ecuaciones elípticas». En Autores, *Problemas resueltos de métodos numéricos*. Madrid: Thomson.



En el capítulo 10.4 de este libro puedes encontrar las ideas clave para el **desarrollo de las EDP elípticas, así como propuestas de implementación en Matlab**. Además, al final del capítulo hay una serie de ejercicios resueltos para clarificar los conceptos trabajados, que incluyen ejemplos con condiciones no Dirichlet.

Cálculo numérico

Cordero, A. & Hueso, J. L. y Torregrosa, J. R. (2004). *Cálculo numérico*. Valencia: Universidad Politécnica de Valencia.



El capítulo 4.5 de este libro las EDP elípticas utilizando el método de diferencias finitas y los métodos iterativos de Jacobi, Gauss-Seidel, sobrerrelajación y bloques. Además, contiene implementaciones del código para resolver los problemas en Matlab. Al final del capítulo, se encuentran una serie de ejercicios resueltos.

Test

1. Una EDP elíptica cumple que $\Delta = B^2 - AC$:

A. Es menor que cero.

B. Es igual a cero.

C. Es mayor que cero.

2. Las condiciones de contorno de tipo Dirichlet en una EDP elíptica dan la solución en:

A. x=a, x=b.

B. y=c, y=d.

C. Todas las anteriores son correctas.

3. Cuando discretizamos una EDP elíptica, utilizamos diferencias centrales sobre:

A. Ninguna variable.

B. Una variable.

C. Las dos variables.

4. El resultado de la discretización permite resolver la EDP elíptica como un:

A. Sistema lineal.

B. Sistema no lineal.

C. Sistema invariante.

5. Dada una EDP elíptica con condiciones de tipo Dirichlet, discretizada con $nx = ny = n$, ¿en cuántos nodos hay que obtener la solución?

A. $(n-1)^2$

B. n^2

C. $(n+1)^2$

6. En el método de Jacobi, ¿a partir de qué valores obtenemos los iterados actuales?

A. A partir de los valores del iterado anterior.

B. A partir de los valores del iterado anterior y de los valores del iterado actual que ya conocemos.

C. A partir de los valores del iterado actual.

7. En el método de Gauss-Seidel, ¿a partir de qué valores obtenemos los iterados actuales?

A. A partir de los valores del iterado anterior.

B. A partir de los valores del iterado anterior y de los valores del iterado actual que ya conocemos.

C. A partir de los valores del iterado actual.

8. En el método SOR, ¿entre qué valores puede estar el factor de relajación?

A. Entre 0 y 1.

B. Valores mayores que 1.

C. Ninguna de las anteriores es correcta.

9. Sea $\hat{u}^{(1)}$ la solución por el método de Gauss-Seidel en la iteración 1 y sea $u^{(0)}$ la estimación inicial. En el método SOR, ¿cómo obtenemos $u^{(1)}$?

A. $u^{(1)} = \omega\, \hat{u}^{(1)} + u^{(0)}$

B. $u^{(1)} = \omega\, \hat{u}^{(1)} - (1-\omega)u^{(0)}$

C. $u^{(1)} = \omega\, \hat{u}^{(1)} + (1-\omega)u^{(0)}$

10. Sean los métodos de Jacobi, Gauss-Seidel y SOR con ω tomando el valor óptimo. ¿Cuál de ellos obtiene la solución en un menor número de iteraciones?

A. Jacobi.

B. Gauss-Seidel.

C. SOR.