MÉTODOS NUMÉRICOS Y PROGRAMACIÓN 2023/2024

Métodos Directos e Iterativos para SEL (PRÁCTICA 4)

1.— Escribir un algoritmo de inversión para matrices triangulares inferiores de orden n y semiancho de banda l. La matriz inicial debe almacenarse en un vector, y su inversa debe almacenarse sobre la matriz inicial.

Solución 1.

La estructura de las matrices será a priori:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & & & & & \\ a_{21} & a_{22} & & & & \\ \vdots & & \ddots & & & \\ a_{1+l,1} & \dots & \dots & a_{1+l,1+l} & & \\ & \ddots & & & \ddots & \\ & & a_{n,n-l} & \dots & \dots & a_{n,n} \end{bmatrix} \quad ; \quad \boldsymbol{A}^{-1} = \boldsymbol{X} = \underbrace{\begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nn} \end{bmatrix}}_{\text{matriz llena}}$$

Dado que $\boldsymbol{X}$ es la inversa de la matriz $\boldsymbol{A}$:

$$\boldsymbol{A}\,\boldsymbol{X} = \boldsymbol{I} \Leftrightarrow \left\{ \begin{array}{c} \boldsymbol{A}\,\boldsymbol{x}^k = \boldsymbol{\delta}^k \\ k = 1, \dots, n \end{array} \right\}, \text{siendo} \quad \boldsymbol{x}^k = \begin{bmatrix} x_{1k} \\ x_{2k} \\ \vdots \\ x_{nk} \end{bmatrix} \quad ; \quad \boldsymbol{\delta}^k = \begin{bmatrix} \delta_{1k} \\ \delta_{2k} \\ \vdots \\ \delta_{nk} \end{bmatrix}$$

Por tanto la obtención de la matriz inversa se puede plantear como la resolución de n sistemas de ecuaciones con matriz A y término independiente $\boldsymbol{\delta}^k$.

$$\left. \begin{array}{l} x_{kk} = 1/a_{kk} \\ x_{ik} = -\left(\displaystyle\sum_{j=k}^{i-1} a_{ij}\, x_{jk} \right) / a_{ii} \quad ; \quad i = k+1, \dots, n \end{array} \right\} k = 1, \dots, n$$

El algoritmo anterior se puede modificar para almacenar la matriz inicial en un vector, y la solución sobre el mismo, aunque estaremos obligados a almacenar la matriz $\boldsymbol{A}$ como matriz triangular completa y no en banda ya que $\boldsymbol{X}$ es llena.

Para almacenar $\boldsymbol{A}$ en un vector:

$$a_{\alpha\beta} \longrightarrow v_\gamma \quad \text{con} \quad \gamma = \frac{\alpha(\alpha-1)}{2} + \beta$$

Introduciendo el cambio:

$$\left. \begin{array}{l} v_{\frac{k(k-1)}{2}+k} = \dfrac{1}{v_{\frac{k(k-1)}{2}+k}} \\ v_{\frac{i(i-1)}{2}+k} = -\left(\displaystyle\sum_{j=max\{k,i-l\}}^{i-1} v_{\frac{i(i-1)}{2}+j}\; v_{\frac{j(j-1)}{2}+k} \right) / v_{\frac{i(i-1)}{2}+i} \quad ; \quad i = k+1, \dots, n \end{array} \right\} k = 1, \dots, n$$

2.— Escribir un algoritmo de resolución para sistemas lineales de la forma $\boldsymbol{K}\,\boldsymbol{u} = \boldsymbol{f}$, siendo $\boldsymbol{K}$ una matriz simétrica y definida positiva, de semiancho de banda s, almacenada en un vector. Utilícese el esquema de almacenamiento y el método de solución más adecuados.

Solución 2.

En este caso y dado que la matriz del sistema es simétrica y definida positiva, el método más adecuado es utilizar una factorización de Cholesky y resolver el sistema $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ de la siguiente forma:

$$\boldsymbol{A} = \boldsymbol{L}\,\boldsymbol{D}\,\boldsymbol{L}^t \longrightarrow \boldsymbol{L}\,\underbrace{(\boldsymbol{D}\,\overbrace{(\boldsymbol{L}^t\,\boldsymbol{x})}^{\boldsymbol{y}})}_{\boldsymbol{z}} = \boldsymbol{b} \longrightarrow \left\{ \begin{array}{lcl} \boldsymbol{L}\,\boldsymbol{z} & = & \boldsymbol{b} \\ & \searrow & \\ \boldsymbol{D}\,\boldsymbol{y} & = & \boldsymbol{z} \\ & \searrow & \\ \boldsymbol{L}^t\,\boldsymbol{x} & = & \boldsymbol{y} \end{array} \right.$$

La factorización en su planteamiento general de una matriz $\boldsymbol{A}$ se plantea de la siguiente forma. Sean:

$$\boldsymbol{A}_k = \begin{bmatrix} a_{11} & \dots & a_{1k} \\ \vdots & & \vdots \\ a_{k1} & \dots & a_{kk} \end{bmatrix} \quad ; \quad \boldsymbol{A}_{k+1} = \left[\begin{array}{c|c} \boldsymbol{A}_k & \boldsymbol{f}_{k+1} \\ \hline \boldsymbol{f}_{k+1}^t & a_{k+1,k+1} \end{array}\right] \quad ; \quad \text{con} \quad \boldsymbol{f}_{k+1}^t = [a_{k+1,1}, \dots, a_{k+1,k}]$$

$$\boldsymbol{L}_k = \begin{bmatrix} l_{11} & \dots & l_{1k} \\ \vdots & & \vdots \\ l_{k1} & \dots & l_{kk} \end{bmatrix} \quad ; \quad \boldsymbol{L}_{k+1} = \left[\begin{array}{c|c} \boldsymbol{L}_k & \boldsymbol{0} \\ \hline \boldsymbol{l}_{k+1}^t & l_{k+1,k+1} \end{array}\right] \quad ; \quad \text{con} \quad \boldsymbol{l}_{k+1}^t = [l_{k+1,1}, \dots, l_{k+1,k}]$$

$$\boldsymbol{D}_k = \begin{bmatrix} d_{11} & & \\ & \ddots & \\ & & d_{kk} \end{bmatrix} \quad ; \quad \boldsymbol{D}_{k+1} = \left[\begin{array}{c|c} \boldsymbol{D}_k & \boldsymbol{0} \\ \hline \boldsymbol{0} & d_{k+1,k+1} \end{array}\right]$$

Si suponemos conocidos $\boldsymbol{L}_k$ y $\boldsymbol{D}_k$ tales que $\boldsymbol{A}_k = \boldsymbol{L}_k\boldsymbol{D}_k\boldsymbol{L}_k^t$, podemos imponer que se cumpla $\boldsymbol{A}_{k+1} = \boldsymbol{L}_{k+1}\boldsymbol{D}_{k+1}\boldsymbol{L}_{k+1}^t$, es decir:

$$\left[\begin{array}{c|c} \boldsymbol{A}_k & \boldsymbol{f}_{k+1} \\ \hline \boldsymbol{f}_{k+1}^t & a_{k+1,k+1} \end{array}\right] = \left[\begin{array}{c|c} \boldsymbol{L}_k\boldsymbol{D}_k\boldsymbol{L}_k^t & \boldsymbol{L}_k\boldsymbol{D}_k\boldsymbol{l}_{k+1} \\ \hline \boldsymbol{l}_{k+1}^t\boldsymbol{D}_k\boldsymbol{L}_k^t & \boldsymbol{l}_{k+1}^t\boldsymbol{D}_k\boldsymbol{l}_{k+1} + l_{k+1,k+1}^2 d_{k+1,k+1} \end{array}\right]$$

Por tanto, se tiene que cumplir:

$$\left\{ \begin{array}{l} \boldsymbol{L}_k\boldsymbol{D}_k\boldsymbol{l}_{k+1} = \boldsymbol{f}_{k+1} \\ \\ a_{k+1,k+1} = \boldsymbol{l}_{k+1}^t\boldsymbol{D}_k\boldsymbol{l}_{k+1} + l_{k+1,k+1}^2 d_{k+1,k+1} \end{array} \right.$$

luego:

- $\boldsymbol{l}_{k+1}$ se obtiene resolviendo el sistema $(\boldsymbol{L}_k\boldsymbol{D}_k)\boldsymbol{l}_{k+1} = \boldsymbol{f}_{k+1}$
- $l_{k+1,k+1} = 1$ de forma arbitraria.
- $d_{k+1,k+1} = a_{k+1,k+1} - \boldsymbol{l}_{k+1}^t \boldsymbol{D}_k \boldsymbol{l}_{k+1}$

Para $k = 1$ se obtiene directamente que $d_{11} = a_{11}$ y $l_{11} = 1$. Por tanto el algoritmo general de factorización para una matriz simétrica será:

$$l_{11} = 1 \quad ; \quad d_{11} = a_{11}$$

do $k = 1, n-1$

$$l_{k+1,i} = a_{k+1,i} - \sum_{j=1}^{i-1} l_{ij}\, l_{k+1,j}; \qquad i = 1, \ldots, k$$

$$l_{k+1,i} = l_{k+1,i}/d_{ii} \qquad i = 1, \ldots, k$$

$$l_{k+1,k+1} = 1$$

$$d_{k+1,k+1} = a_{k+1,k+1} - \sum_{j=1}^{k} l_{k+1,j}^2\, d_{jj}$$

enddo

Y la resolución del sistema:

$$z_i = b_i - \sum_{j=1}^{i-1} l_{ij}\, z_j \qquad ; \quad i = 1, \ldots, n$$

$$y_i = z_i/d_{ii} \qquad ; \quad i = 1, \ldots, n$$

$$x_i = y_i - \sum_{j=1+1}^{n} l_{ji}\, x_j \quad ; \quad i = n, \ldots, 1 \qquad \text{(Hacia atrás)}$$

Dado que la diagonal de $\boldsymbol{L}$ y los elementos de $\boldsymbol{A}$ no se vuelven a utilizar, es posible almacenar la información de las matrices $\boldsymbol{L}$ y $\boldsymbol{D}$ sobre $\boldsymbol{A}$. Lo mismo ocurre con los vectores $\boldsymbol{z}$, $\boldsymbol{y}$ y $\boldsymbol{x}$ y el término independiente del sistem $\boldsymbol{b}$.

Teniendo en cuenta lo anterior, la factorización y la resolución del sistema resultarían:

do $k = 1, \ldots, n-1$

$$a_{k+1,i} = a_{k+1,i} - \sum_{j=1}^{i-1} a_{ij}\, a_{k+1,j}; \qquad i = 2, \ldots, k$$

$$a_{k+1,i} = a_{k+1,i}/a_{ii} \qquad i = 1, \ldots, k$$

$$a_{k+1,k+1} = a_{k+1,k+1} - \sum_{j=1}^{k} a_{k+1,j}^2\, a_{jj}$$

enddo

$$b_i = b_i - \sum_{j=1}^{i-1} a_{ij}\, b_j \quad ; \quad i = 2, \ldots, n$$

$$b_i = b_i / a_{ii} \quad ; \quad i = 1, n$$

$$b_i = b_i - \sum_{j=1+1}^{n} a_{ji}\, b_j \quad ; \quad i = n-1, \ldots, 1 \qquad \text{(Hacia atrás)}$$

Se puede comprobar además que el semiancho de banda de la matriz $\boldsymbol{L}$ es el mismo que el de $\boldsymbol{A}$. Por tanto, tanto antes como después de la factorización, los elementos a_{ij} fuera de la banda son nulos. Modificando los bucles para no operar con esos elementos obtenemos:

do $k = 1, n-1$

$$a_{k+1,i} = a_{k+1,i} - \sum_{j=max\{i-s,k+1-s,1\}}^{i-1} a_{ij}\, a_{k+1,j} \quad ; \quad i = max\left\{k+1-s+1, 2\right\}, \ldots, k$$

$$a_{k+1,i} = a_{k+1,i} / a_{ii} \quad ; \quad i = max\left\{k+1-s, 1\right\}, \ldots, k$$

$$a_{k+1,k+1} = a_{k+1,k+1} - \sum_{j=max\{k+1-s,1\}}^{k} a_{k+1,j}^2\, a_{jj}$$

enddo

$$b_i = b_i - \sum_{j=max\{i-s,1\}}^{i-1} a_{ij}\, b_j \quad ; \quad i = 2, \ldots, n$$

$$b_i = b_i / a_{ii} \quad ; \quad i = 1, \ldots, n$$

$$b_i = b_i - \sum_{j=1+1}^{min\{i+s,n\}} a_{ji}\, b_j \quad ; \quad i = n-1, \ldots, 1 \qquad \text{(Hacia atrás)}$$

Además, podemos almacenar a su vez la matriz en banda $\boldsymbol{A}$ en un vector de forma que los elementos $a_{\alpha,\beta}$ se almacenen en un vector v_γ con $\gamma = (\alpha-1)(1+s) + (\beta - \alpha + 1 + s) = \beta + \alpha\, s$. De esta forma:

do $k = 1, n-1$

$$v_{(k+1)s+i} = v_{(k+1)s+i} - \sum_{j=max\{i-s,k+1-s,1\}}^{i-1} v_{is+j}\, v_{(k+1)s+j} \quad ; \quad i = max\left\{k+1-s+1, 2\right\}, \ldots, k$$

$$v_{(k+1)s+i} = v_{(k+1)s+i} / v_{is+i} \quad ; \quad i = max\left\{k+1-s, 1\right\}, \ldots, k$$

$$v_{(k+1)s+(k+1)} = v_{(k+1)s+(k+1)} - \sum_{j=max\{k+1-s,1\}}^{k} v_{(k+1)s+j}^2\, v_{js+j}$$

enddo

$$b_i = b_i - \sum_{j=max\{i-s,1\}}^{i-1} v_{is+j}\, b_j \quad ; \quad i = 2, \ldots, n$$

$$b_i = b_i / v_{is+i} \quad ; \quad i = 1, \ldots, n$$

$$b_i = b_i - \sum_{j=1+1}^{min\{i+s,n\}} v_{js+i}\, b_j \quad ; \quad i = n-1, \ldots, 1 \qquad \text{(Hacia atrás)}$$

3.— Generalizar el algoritmo anterior para la resolución de m sistemas de ecuaciones con la misma matriz y diferentes términos independientes $\{\boldsymbol{f}^1, \boldsymbol{f}^2, \ldots, \boldsymbol{f}^m\}$, hallándose éstos almacenados en un vector.

Solución 3.

Nos piden resolver m sistemas de ecuaciones de la forma:

$$\boldsymbol{A}\,\boldsymbol{x}^p = \boldsymbol{b}^p \qquad p = 1, \ldots, m$$

Dado que la matriz es la misma, la factorización es igual a la del problema anterior y solamente se realiza una vez. Luego solo es necesario repetir m veces la solución de sistemas.

Si almacenamos los m términos independientes en un único vector $\boldsymbol{g}$, las componentes f_α^p se almacenarán en g_γ con $\gamma = (p-1)n + \alpha$. El algoritmo de resolución sería por tanto:

do $p = 1, \ldots, m$

$$b_{(p-1)n+i} = b_{(p-1)n+i} - \sum_{j=max\{i-s,1\}}^{i-1} v_{is+j}\, b_{(p-1)n+j} \quad ; \quad i = 2, \ldots, n$$

$$b_{(p-1)n+i} = b_{(p-1)n+i} / v_{is+i} \quad ; \quad i = 1, \ldots, n$$

$$b_{(p-1)n+i} = b_{(p-1)n+i} - \sum_{j=1+1}^{min\{i+s,n\}} v_{js+i}\, b_{(p-1)n+j} \quad ; \quad i = n-1, \ldots, 1 \qquad \text{(Hacia atrás)}$$

enddo

4.— Para analizar un cierto problema de ingeniería es preciso resolver sistemas de ecuaciones lineales $\boldsymbol{Ax} = \boldsymbol{b}$, donde la matriz $\boldsymbol{A}$ es simétrica, definida positiva y tiene la forma:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & & & & & & \\ a_{21} & a_{22} & & & & SIM & \\ a_{31} & 0 & a_{33} & & & & \\ a_{41} & 0 & 0 & a_{44} & & & \\ a_{51} & 0 & 0 & 0 & a_{55} & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \\ a_{n1} & 0 & \ldots & \ldots & \ldots & 0 & a_{nn} \end{bmatrix}$$

$1000 \leq n \leq 5000$
$|a_{ii}| >> |a_{ij}| \;\; \forall j \neq i$

Para resolver el sistema de ecuaciones se quiere utilizar un método directo, almacenando las operaciones intermedias sobre la propia matriz. Se pide:

a) ¿Cuál es el esquema de almacenamiento más económico que se puede usar en estas condiciones? ¿Por qué?

b) ¿Qué método directo conviene utilizar en estas condiciones?¿Por qué?

c) ¿Podría ser ventajoso, en cuanto a tiempo de cálculo o almacenamiento, utilizar un algoritmo iterativo?¿Cuál?¿Por qué?

d) ¿Podría ser ventajoso, en cuanto a tiempo de cálculo o almacenamiento, utilizar un algoritmo semi-iterativo?¿Cuál?¿Por qué?

e) Si se renumeran adecuadamente las ecuaciones y/o las incógnitas, ¿es posible reducir el coste computacional y el almacenamiento asociados a la utilización de un método directo?¿Cómo?

Solución 4.a

Si se utiliza un método directo para resolver el problema (Gauss para matrices simétricas o factorización de Cholesky) los elementos nulos dejarán de ser nulos durante el proceso de resolución por lo que será necesario almacenar los ceros. Por tanto hay que almacenar la matriz como simétrica, bien la parte triangular inferior o la parte triangular superior con $n(n+1)/2$ elementos.

Solución 4.b

Tanto Gauss para matriz simétrica como factorización de Cholesky serían adecuados. En general se preferirá la factorización de Cholesky. Ambos métodos funcionarán ya que la matriz es definida positiva.

Solución 4.c

La matriz es diagonalmente dominante ($|a_{ii}| >> |a_{ij}| \ \forall j \neq i$) por lo que los métodos de Jacobi y Gauss-Seidel convergerán.

Es posible que el uso de estos métodos sea ventajoso porque muchos elementos de la matriz son nulos por lo que no es necesario tenerlos en cuenta. El coste computacional por iteración será entonces T(n) frente al coste computacional T(n^3) de un método directo para matriz llena.

Solución 4.d

La matriz es definida positiva por lo que el método de Gradiente Conjugado convergerá a la solución. Por el mismo motivo que se apuntó en el apartado anterior, el coste por iteración será T(n). Como máximo se realizarán n iteraciones, luego el coste total de un método semi-iterativo sería de T(n^2), que es siempre ventajoso.

Solución 4.e

Podemos renumerar las ecuaciones y las incógnitas de la siguiente forma:

$$\begin{bmatrix} a_{nn} & & & & & & a_{n1} \\ & \ddots & & & & & \vdots \\ & & a_{55} & & & & a_{51} \\ & & & a_{44} & & & a_{41} \\ & & & & a_{33} & & a_{31} \\ & SIM & & & & a_{22} & a_{21} \\ & & & & & & a_{11} \end{bmatrix} \begin{bmatrix} x_n \\ \vdots \\ x_5 \\ x_4 \\ x_3 \\ x_2 \\ x_1 \end{bmatrix} = \begin{bmatrix} b_n \\ \vdots \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \end{bmatrix}$$

De esta forma el sistema de ecuaciones sigue siendo el mismo pero la matriz del sistema para a ser simétrica en perfil, es decir, únicamente hay que almacenar $2n-1$ componentes y el tiempo de computación pasa a ser T(n).

5.— Deducir la condición suficiente para que el algoritmo iterativo

$$\boldsymbol{B}^{k+1} = \boldsymbol{B}^k[2\,\boldsymbol{I} - \boldsymbol{A}\,\boldsymbol{B}^k]; \qquad k = 0, \ldots$$

converja a la inversa de la matriz $\boldsymbol{A}$, siendo $\boldsymbol{B}^0$ una aproximación inicial a $\boldsymbol{A}^{-1}$ tal que:

$$\boldsymbol{A}\,\boldsymbol{B} = \boldsymbol{I} + \boldsymbol{E}; \qquad (\boldsymbol{E} = \text{matriz de error})$$

Solución 5.

Definimos el error que se comete en la iteración k como:

$$\boldsymbol{E}_k = \boldsymbol{A}\,\boldsymbol{B}_k - \boldsymbol{I}$$

por tanto:

$$\begin{aligned}
\boldsymbol{E}_{k+1} &= \boldsymbol{A}\,\boldsymbol{B}_{k+1} - \boldsymbol{I} = \boldsymbol{A}\,\boldsymbol{B}_k[2\boldsymbol{I} - \boldsymbol{A}\,\boldsymbol{B}_k] - \boldsymbol{I} \\
&= (\boldsymbol{E}_k + \boldsymbol{I})[2\boldsymbol{I} - (\boldsymbol{E}_k + \boldsymbol{I})] - \boldsymbol{I} \\
&= (\boldsymbol{E}_k + \boldsymbol{I})[\boldsymbol{E}_k + \boldsymbol{I}] - \boldsymbol{I} \\
&= -(\boldsymbol{E}_k)^2 + \boldsymbol{E}_k - \boldsymbol{E}_k + \boldsymbol{I} - \boldsymbol{I} \\
&= -(\boldsymbol{E}_k)^2
\end{aligned}$$

así, el error:

$$\begin{aligned}
&\boldsymbol{E}_0 = \boldsymbol{E} \\
&\boldsymbol{E}_1 = -(\boldsymbol{E}_0)^2 = -\boldsymbol{E}^2 \\
&\boldsymbol{E}_2 = -(\boldsymbol{E}_1)^2 = -\boldsymbol{E}^4 \\
&\boldsymbol{E}_3 = -(\boldsymbol{E}_2)^2 = -\boldsymbol{E}^8 \\
&\vdots \\
&\boldsymbol{E}_k = -(\boldsymbol{E}_{k-1})^2 = -\boldsymbol{E}^{(2^k)}
\end{aligned}$$

La condición de convergencia será:

$$\lim_{k\to\infty} \boldsymbol{B}_k = \boldsymbol{A}^{-1} \Leftrightarrow \lim_{k\to\infty} \boldsymbol{E}_k = \boldsymbol{0} \Leftrightarrow \rho(\boldsymbol{E}) < 1$$

siendo $\rho(\boldsymbol{E})$ el radio espectral de la matriz $\boldsymbol{E}$
Podemos comprobar además el orden de convergencia:

$$\begin{aligned}(\boldsymbol{B}_{k+1} - \boldsymbol{A}^{-1}) &= \boldsymbol{A}^{-1}(\boldsymbol{A}\,\boldsymbol{B}_{k+1} - \boldsymbol{I}) = \boldsymbol{A}^{-1}\boldsymbol{E}_{k+1} = -\boldsymbol{A}^{-1}(\boldsymbol{E}_k)^2 \\ &= -\boldsymbol{A}^{-1}(\boldsymbol{A}\,\boldsymbol{B}_k - \boldsymbol{I})^2 = -\boldsymbol{A}^{-1}(\boldsymbol{A}(\boldsymbol{B}_k - \boldsymbol{A}^{-1}))^2 \\ &= -\boldsymbol{A}^{-1}\,\boldsymbol{A}(\boldsymbol{B}_k - \boldsymbol{A}^{-1})\boldsymbol{A}(\boldsymbol{B}_k - \boldsymbol{A}^{-1}) = -(\boldsymbol{B}_k - \boldsymbol{A}^{-1})\boldsymbol{A}(\boldsymbol{B}_k - \boldsymbol{A}^{-1})\end{aligned}$$

$$\Rightarrow ||\boldsymbol{B}_{k+1} - \boldsymbol{A}^{-1}|| \leq ||\boldsymbol{A}||\,||\boldsymbol{B}_k - \boldsymbol{A}^{-1}||^2 \longrightarrow \text{Convergencia Cuadrática}$$

6.— Para resolver el sistema de ecuaciones:

$$\begin{pmatrix} 4 & 1 \\ 1 & 4 \end{pmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 6 \\ 9 \end{Bmatrix}$$

Se desea utilizar el método de Gauss-Seidel sobrerrelajado. Se pide:

a) Estudiar la convergencia del algoritmo en función del coeficiente de sobrerrelajación α usado (α= constante en todas las iteraciones).

b) ¿Existe algún valor óptimo del coeficiente de sobrerrelajación (constante en todas las iteraciones) para que la convergencia sea más rapida? En caso afirmativo, hallarlo. (Sugerencia: representar gráficamente el radio espectral de la matriz correspondiente en función del coeficiente de sobrerrelajación).

c) Realizar las cinco primeras iteraciones para $\alpha = 1$ y para $\alpha = \frac{32}{31}$, partiendo de la aproximación inicial $x_1^0 = 0,\ x_2^0 = 0$.

d) En la práctica, ¿podría realizarse un análisis como el anterior para un sistema de varios miles de ecuaciones?¿Por qué?

Solución 6.a

El esquema del algoritmo de Gauss-Seidel sobrerrelajado será:

$$\boldsymbol{x}k + 1 = \boldsymbol{x}^k + \alpha\,\boldsymbol{C}^{-1}(\boldsymbol{b} - \boldsymbol{A}\,\boldsymbol{x}^k) \quad ; \quad \text{con} \quad \boldsymbol{C} = \begin{bmatrix} 4 & 0 \\ 1 & 4 \end{bmatrix}$$

Definimos el error en la iteración k como :

$$\boldsymbol{e}^k = \boldsymbol{x} - \boldsymbol{x}^k \quad \longrightarrow \quad \boldsymbol{e}^{k+1} = \left[\boldsymbol{I} - \alpha\,\boldsymbol{C}^{-1}\boldsymbol{A}\right]\boldsymbol{e}^k$$

por lo que la condición de convergencia es $\rho(\boldsymbol{I} - \alpha\,\boldsymbol{C}^{-1}\boldsymbol{A}) < 1$, y cuanto menor sea, mayor será la velocidad de convergencia.

Para estudiar la convergencia, por tanto, estudiamos los autovalores de $(\boldsymbol{I} - \alpha\,\boldsymbol{C}^{-1}\boldsymbol{A})$ en función de α

$$(\boldsymbol{I} - \alpha\,\boldsymbol{C}^{-1}\boldsymbol{A})\boldsymbol{u} = \lambda\boldsymbol{u} \quad \Leftrightarrow \quad det(\boldsymbol{I} - \alpha\,\boldsymbol{C}^{-1}\boldsymbol{A} - \lambda\,\boldsymbol{I}) = 0 \Leftrightarrow$$

$$\Leftrightarrow det((1-\lambda)\boldsymbol{C}-\alpha\boldsymbol{A})=0 \quad \Leftrightarrow \quad det(\alpha\boldsymbol{A}+(\lambda-1)\boldsymbol{C})=0 \Leftrightarrow$$

$$\Leftrightarrow \begin{vmatrix} 4\alpha+(\lambda-1)4 & 1\alpha+(\lambda-1)0 \\ 1\alpha+(\lambda-1)1 & 4\alpha+(\lambda-1)4 \end{vmatrix} = \begin{vmatrix} 4(\alpha+\lambda-1) & \alpha \\ (\alpha+\lambda-1) & 4(\alpha+\lambda-1) \end{vmatrix} = 0 \Leftrightarrow$$

$$\Leftrightarrow 16(\alpha+\lambda-1)^2-\alpha(\alpha+\lambda-1)=0 \quad \Leftrightarrow \quad \begin{cases} (\alpha+\lambda-1)=0 \quad \longrightarrow \quad \lambda=1-\alpha \\ \text{ó} \\ 16(\alpha+1-1)-\alpha=0 \quad \longrightarrow \quad \lambda=1-\frac{15\alpha}{16} \end{cases}$$

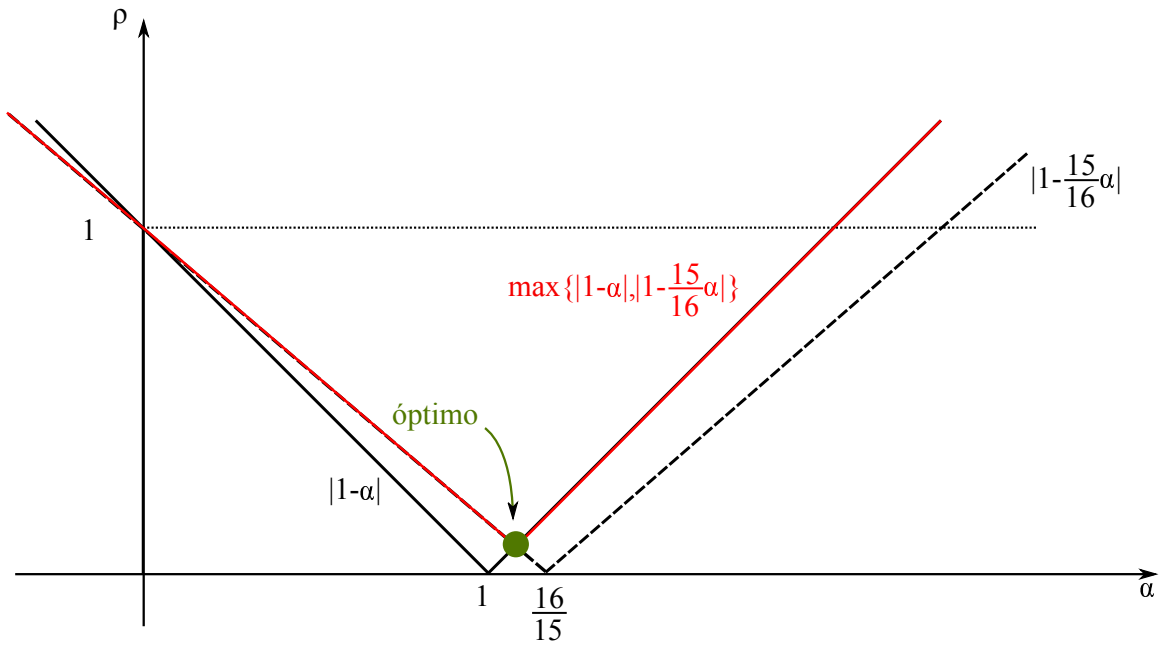
$$\rho(\boldsymbol{I}-\alpha\boldsymbol{C}^{-1}\boldsymbol{A})=max\left\{|1-\alpha|,\left|1-\frac{15\alpha}{16}\right|\right\}$$

Por lo que, para que el algoritmo converja.

$$\left\{\begin{array}{c} |1-\alpha|<1 \Leftrightarrow -1<1-\alpha<1 \Leftrightarrow -2<-\alpha<0 \\ y \\ \left|1-\frac{15\alpha}{16}\right|<1 \Leftrightarrow -1<|1-\frac{15\alpha}{16}<1 \Leftrightarrow -2<1-\frac{15\alpha}{16}<0 \end{array}\right\} \Leftrightarrow \left\{\begin{array}{c} \alpha>0 \\ \alpha<2 \\ \alpha<\frac{32}{15} \end{array}\right\}$$

Luego el algoritmo es convergente si $0<\alpha<2$.

Solución 6.b



El valor óptimo de α es aquel que minimiza el el radio espectral $\rho(\boldsymbol{I}-\alpha\boldsymbol{C}^{-1}\boldsymbol{A})$, es decir:

$$\alpha-1=1-\frac{15\alpha}{16} \quad \Leftrightarrow \quad \alpha\left(1+\frac{15}{16}\right)=2 \quad \Leftrightarrow \quad \alpha=\frac{32}{31}=\alpha_{opt}$$

Solución 6.c

La forma más práctica de realizar los cálculos es la siguiente:

$$\left\{ \begin{array}{l} \boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha \boldsymbol{s}^k \\ \boldsymbol{C}\,\boldsymbol{s}^k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^k \end{array} \right\} \Rightarrow \left\{ \begin{array}{l} \left\{ \begin{array}{c} x_1^{k+1} \\ x_2^{k+1} \end{array} \right\} = \left\{ \begin{array}{c} x_1^k \\ x_2^k \end{array} \right\} + \alpha \left\{ \begin{array}{c} s_1^k \\ s_2^k \end{array} \right\} \\ \\ \left[\begin{array}{cc} 4 & 0 \\ 1 & 4 \end{array} \right] \left\{ \begin{array}{c} s_1^k \\ s_2^k \end{array} \right\} = \left\{ \begin{array}{c} 6 \\ 9 \end{array} \right\} - \left[\begin{array}{cc} 4 & 1 \\ 1 & 4 \end{array} \right] \left\{ \begin{array}{c} x_1^k \\ x_2^k \end{array} \right\} \end{array} \right.$$

$$\alpha = 1 \Rightarrow \left\{ \begin{array}{l} x_1^{k+1} = -x_2^k/4 + 3/2 \\ x_2^{k+1} = x_2^k/16 + 15/8 \end{array} \right.$$

$$\alpha = 32/31 \Rightarrow \left\{ \begin{array}{l} x_1^{k+1} = 48/31 - 1/31\, x_1^k - 8/31\, x_2^k \\ x_2^{k+1} = 60/31 + 2/31\, x_2^k \end{array} \right.$$

Realizando las primeras 5 iteraciones partiendo de la solución inicial proporcionada:

k	x_1^k ($\alpha = 1$)	x_2^k ($\alpha = 1$)	x_1^k ($\alpha = 32/31$)	x_2^k ($\alpha = 32/31$)
0	0.000000000	0.000000000	0.000000000	0.000000000
1	1.500000000	1.875000000	1.548387097	1.935483871
2	1.031250000	1.992187500	0.998959417	1.997918835
3	1.001953125	1.999511719	1.000570642	1.999932866
4	1.000122070	1.999969482	0.999998917	1.999997834
5	1.000007629	1.999998093	1.000000594	1.999999930

Observamos que los resultados son muy similares, y en ambos casos el método converge a la solución $\boldsymbol{x} = \{1, 2\}$.

Se puede observar también que para el α óptimo la convergencia es cas cuadrática aunque en el caso de $\alpha = 1$ la velocidad es también rápida.

Solución 6.d

Para grandes sistemas no es posible realizar un estudio analítico como el anterior porque no es posible obtener analíticamente el radio espectral de la matriz. Pero es posible realizar algunas iteraciones para distintos valores de α para un sistema con la misma matriz y solución conocida (por ejemplo $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$) con el fin de obtener experimentalmente un valor de α que produzca una velocidad de convergencia razonablemente alta.

7.— Sea una viga continua formada por n vanos. Se numeran los vanos y los apoyos correlativamente, de forma que el vano $e \in \{1, \ldots, n\}$ se extiende desde el apoyo $e-1$ al apoyo e. Sea E el módulo de elasticidad del material, y sean I_e y L_e el momento de inercia de la sección y la longitud correspondientes al vano e-ésimo, respectivamente. Dados los momentos $\{M_i\}_{i=0,\ldots,n}$ que actúan sobre los apoyos se desea calcular los correspondientes giros $\{\omega_i\}_{i=0,\ldots,n}$ que se producen. Se pide:

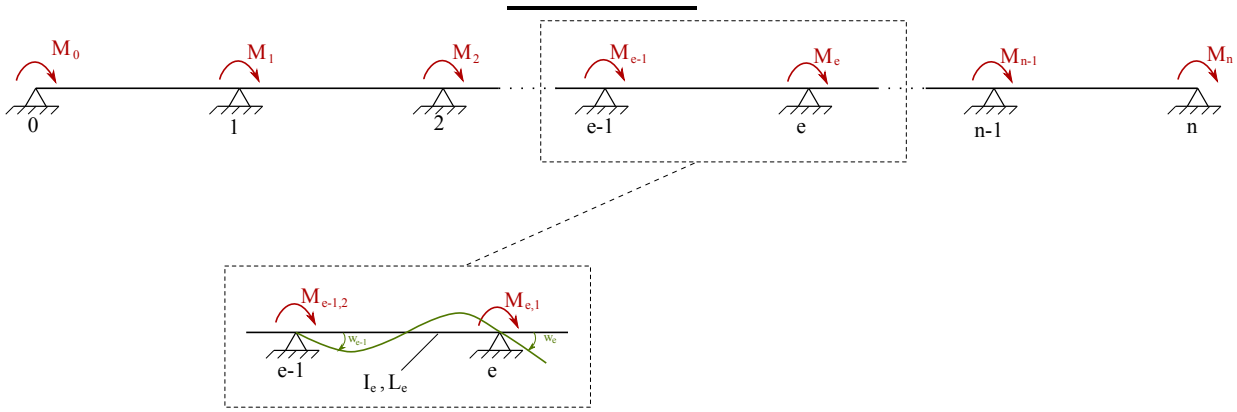
a) Plantear el sistema de ecuaciones lineales que es preciso resolver para calcular los giros.

b) Verificar que la matriz de coeficientes del sistema anterior es simétrica y definida positiva.

c) Proponer y desarrollar completamente el método directo que se considere más adecuado para resolver el sistema.

d) Implementar el método seleccionado en un programa FORTRAN que permita resolver este tipo de probelmas.

e) Plantear la solución del sistema mediante el método iterativo de Gauss-Seidel. ¿Se puede asegurar que este método convergerá? Interpretar el funcionamiento del método desde el punto de vista estructural.



Solución 7.a

$$\left\{ \begin{array}{c} \omega_{e-1} = M_{e-1,2}\dfrac{L_e}{3EI_e} - M_{e,1}\dfrac{L_e}{6EI_e} \\ \omega_e = M_{e,1}\dfrac{L_e}{3EI_e} - M_{e-1,2}\dfrac{L_e}{6EI_e} \end{array} \right\} \Leftrightarrow$$

$$\Leftrightarrow \left\{ \begin{array}{c} \omega_{e-1} \\ \omega_e \end{array} \right\} = \frac{L_e}{6EI_e} \left[\begin{array}{cc} 2 & -1 \\ -1 & 2 \end{array} \right] \left\{ \begin{array}{c} M_{e-1,2} \\ M_{e,1} \end{array} \right\}$$

Luego:

$$\Leftrightarrow \left\{ \begin{array}{c} M_{e-1,2} \\ M_{e,1} \end{array} \right\} = \frac{EI_e}{L_e} \left[\begin{array}{cc} 4 & 2 \\ 2 & 4 \end{array} \right] \left\{ \begin{array}{c} \omega_{e-1} \\ \omega_e \end{array} \right\}$$

Por equilibrio de momentos:

$$\left\{ \begin{array}{l} M_0 = M_{1,2} \\ \\ M_i = M_{i,1} + M_{i,2} \quad ; \quad i = 1, \ldots, n-1 \\ \\ M_n = M_{n,1} \end{array} \right.$$

Por tanto:

$$\overbrace{\begin{bmatrix} 4k_1 & 2k_1 & & & & & \\ 2k_1 & 4(k_1+k_2) & 2k_2 & & & & \\ & 2k_2 & 4(k_2+k_3) & 2k_3 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 2k_{n-1} & 4(k_{n-1}+k_n) & 2k_n & \\ & & & & 2k_n & 4k_n \end{bmatrix}}^{\boldsymbol{K}} \overbrace{\begin{Bmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \\ \vdots \\ \omega_{n-1} \\ \omega_n \end{Bmatrix}}^{\boldsymbol{\omega}} = \overbrace{\begin{Bmatrix} M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_{n-1} \\ M_n \end{Bmatrix}}^{\boldsymbol{M}}$$

con $k_e = \frac{EI_e}{L_e}; e = 1, \ldots, n$.

Solución 7.b

La matriz $\boldsymbol{K}$ es simétrica como ya se ha visto en el apartado anterior.
Demostraremos ahora que es además definida positiva.

$$\boldsymbol{v}^t\,\boldsymbol{K}\,\boldsymbol{v} = \sum_{e=1}^{n} \begin{Bmatrix} v_{e-1} & v_e \end{Bmatrix} \overbrace{\begin{bmatrix} 4k_e & 2k_e \\ 2k_e & 4k_e \end{bmatrix}}^{\boldsymbol{K}_e} \begin{Bmatrix} v_{e-1} \\ v_e \end{Bmatrix}$$

dado que las matrices de barra $\boldsymbol{K}_e$ son todas definidas positivas pues sus autovalores son $\lambda_1 = 6k_e$ y $\lambda_2 = 2k_e$, por tanto:

$$\boldsymbol{v}^t\,\boldsymbol{K}\,\boldsymbol{v} \geq 0 \quad \forall \boldsymbol{v}$$

Por tanto como mínimo $\boldsymbol{K}$ es semi-definida positiva.
Para el caso $\boldsymbol{v}^t\,\boldsymbol{K}\,\boldsymbol{v} = 0$, esto ocurre si y solo si:

$$\begin{Bmatrix} v_{e-1} & v_e \end{Bmatrix} \begin{bmatrix} 4k_e & 2k_e \\ 2k_e & 4k_e \end{bmatrix} \begin{Bmatrix} v_{e-1} \\ v_e \end{Bmatrix} \quad \Leftrightarrow \quad v_{e-1} = v_e = 0 \quad ; \quad e = 1, \ldots, n$$

$$\Rightarrow \boldsymbol{v}^t\,\boldsymbol{K}\boldsymbol{v} > 0 \quad \forall \boldsymbol{v} \neq \boldsymbol{0}$$

Por tanto $\boldsymbol{K}$ es Definida Positiva.

Solución 7.c

Teniendo en cuenta que la matriz del sistema es simétrica, tridiagonal y definida positiva, el método más adecuado es la factorización de Cholesky adaptada a matrices tridiagonales.
Si factorizamos $\boldsymbol{K} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^t$, las matrices conservarán el ancho de banda, por tanto:

$$\boldsymbol{L} = \begin{bmatrix} 1 & & & & \\ l_1 & 1 & & & \\ & l_2 & 1 & & \\ & & \ddots & \ddots & \\ & & & l_n & 1 \end{bmatrix} \qquad \boldsymbol{D} = \begin{bmatrix} d_0 & & & & \\ & d_1 & & & \\ & & d_2 & & \\ & & & \ddots & \\ & & & & d_n \end{bmatrix}$$

Realizando el producto de los coeficientes de las tres matrices:

$$\boldsymbol{LDL}t = \begin{bmatrix} d_0 & & & & \\ l_1 d_0 & l_1 d_0 l_1 + d_1 & & & SIM \\ & l_2 d_1 & l_2 d_1 l_2 + d_2 & & \\ & & \ddots & \ddots & \\ & & & l_n d_{n-1} & l_n d_{d-1} l_n + d_n \end{bmatrix}$$

igualando a los coeficientes de la matriz $\boldsymbol{K}$:

$$\left|\begin{array}{lll} d_0 = 4\,k_1 & & \\ l_1\,d_0 = 2\,k_1 & \longrightarrow & l_1 = 2\,k_1/d_0 \\ l_1\,d_0\,l_1 + d_1 = 4(k_1+k_2) & \longrightarrow & d_1 = 4(k_1+k_2) - l_1\,d_0\,l_1 \\ l_2\,d_1 = 2\,k_2 & \longrightarrow & l_2 = 2\,k_2/d_1 \\ l_2\,d_1\,l_2 + d_2 = 4(k_2+k_3) & \longrightarrow & d_2 = 4(k_2+k_3) - l_2\,d_1\,l_2 \\ \vdots & & \\ l_{n-1}\,d_{n-2} = 2\,k_{n-1} & \longrightarrow & l_{n-1} = 2\,k_{n-1}/d_{n-2} \\ l_{n-1}\,d_{n-2}\,l_{n-1} + d_{n-1} = 4(k_{n-1}+k_n) & \longrightarrow & d_{n-1} = 4(k_{n-1}+k_n) - l_{n-1}\,d_{n-2}\,l_{n-1} \\ l_n\,d_{n-1} = 2\,k_n & \longrightarrow & l_n = 2\,k_n/d_{n-1} \\ l_n\,d_{n-1}\,l_n + d_n = 4\,k_n & \longrightarrow & d_n = 4\,k_n - l_n\,d_{n-1}\,l_n \end{array}\right.$$

Por lo tanto el algoritmo de factorización será:

$$\begin{array}{l} d_0 = 4\,k_1 \\ \text{do } \; i = 1, n-1 \\ \quad l_i = 2\,k_i/d_{i-1} \\ \quad d_i = 4(k_i + k_{i+1}) - l_i\,d_{i-1}\,l_i \\ \text{enddo} \\ l_n = 2\,k_n/d_{n-1} \\ d_n = 4\,k_n - l_n\,d_{n-1}\,l_n \end{array}$$

La solución del sistema se lleva a cabo siguiendo el esquema siguiente:

$$\boldsymbol{K} = \boldsymbol{L}\,\boldsymbol{D}\,\boldsymbol{L}^t \longrightarrow \boldsymbol{L}\,\underbrace{(\boldsymbol{D}\,\overbrace{(\boldsymbol{L}^t\,\boldsymbol{\omega})}^{\boldsymbol{y}})}_{\boldsymbol{z}} = \boldsymbol{u} \longrightarrow \left\{\begin{array}{lcl} \boldsymbol{L}\,\boldsymbol{z} & = & \boldsymbol{u} \\ & \searrow & \\ \boldsymbol{D}\,\boldsymbol{y} & = & \boldsymbol{z} \\ & \searrow & \\ \boldsymbol{L}^t\,\boldsymbol{\omega} & = & \boldsymbol{y} \end{array}\right.$$

Utilizando únicamente un único vector, el algoritmo resulta:

$$\begin{array}{lll} \omega_i = \omega_i - l_i\,\omega_{i-1} & ; \quad i = 1, \ldots, n & \\ \omega_i = \omega_i / d_i & ; \quad i = 0, \ldots, n & \\ \omega_i = \omega_i - l_{i+1}\,\omega_{i+1} & ; \quad i = n-1, \ldots, 0 & \text{(Hacia atrás)} \end{array}$$

Solución 7.d

```
      IMPLICIT REAL*4 (A-H,O-Z)
      PARAMETER(MXN=100,MXKP=4*MXN+2)
      INTEGER*4 N
      DIMENSIONWORK(MXKP)

1     WRITE(6,2(A,$)')'NUMERO DE VANOS :'
      READ(5,*)N
      IF(N.GT.MXN)CALL ERRORFATAL('NUMERO EXCESIVO DE VANOS(N > MXN)'

      KPLIBRE=1
      CALL DIMENSIONAMIENTODINAMICO(MXKP,KPLIBRE,N,KPK)
      CALL DIMENSIONAMIENTODINAMICO(MXKP,KPLIBRE,N,KPL)
      CALL DIMENSIONAMIENTODINAMICO(MXKP,KPLIBRE,N+1,KPD)
      CALL DIMENSIONAMIENTODINAMICO(MXKP,KPLIBRE,N+1,KPW)

      CALL LEERDATOS(N,WORK(KPK),WORK(KPW)
      CALL RESOLVERSISTEMA(N,WORK(KPK),WORK(KPL),WORK(KPD),WORK(KPW))
      CALL ESCRIBIRRESULTADOS(N,WORK(KPW))

      END
      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      SUBROUTINE LEERDATOS(N,RK,W)
      IMPLICITREAL*8(A-H,O-Z)
      DIMENSION RK(N),W(0:N)

      DO I=1,N
        WRITE(6,100)I
100     FORMAT('VANO',I5,'- - - - - -> RIGIDEZ EI/L =',$)
        READ(5,*)RK(I)
      ENDDO

      DO I=1,N
        WRITE(6,110)I
110     FORMAT('NODO',I5,'- - - - - -> MOMENTO =',$)
        READ(5,*)W(I)
      ENDDO

      RETURN
      END
      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

```
      SUBROUTINE RESOLVERSISTEMA(N,RK,RL,RD,W)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION RK(N),RL(N),RD(0:N),W(0:N)

      I=0
        RD(I)=4,0D+00*(RK(I+1)
      DO I=1,N-1
        RL(I)=2,0D+00*RK(I)/RD(I-1)
        RD(I)=4,0D+00*(RK(I)+RK(I+1))-RL(I)*RD(I-1)*RL(I)
      ENDDO
      I=N
        RL(I)=2,0D+00*RK(I)/RD(I-1)
        RD(I)=4,0D+00*(RK(I)-RL(I)*RD(I-1)*RL(I)

      DO I=1,N
        W(I)=W(I)-RL(I)*W(I-1)
      ENDDO
      DO I=0,N
        W(I)=W(I)/RD(I)
      ENDDO
      DO I=N-1,0,-1
      W(I)=W(I)-RL(I+1)*W(I+1)
      ENDDO

      RETURN
      END
      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      SUBROUTINE ESCRIBIRRESULTADOS(N,W)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION W(0:N)

      WRITE(6,200)
  200 FORMAT('GIROSENLOSNODOS',/
     .       '====================',/
     .       'NODO GIRO(RADIANES)',/
     .       '====================',/

      DO I=0,N
        WRITE(6,210)I,W(I)
  210 FORMAT(1X,I10,D15,6)
      ENDDO

      RETURN
      END
      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

```
      SUBROUTINE DIMENSIONAMIENTODINAMICO(MXKP,KPLIBRE,NCOMPONENTES,KP

      KP = KPLIBRE
      KPLIBRE = KPLIBRE + NCOMPONENTES
      IF(KPLIBRE.GT.MXKP + 1)CALL ERRORFATAL('MEMORIA INSUFICIENTE')

      RETURN
      END
      - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      SUBROUTINE ERRORFATAL(MENSAJE)
      CHARACTER * (*)MENSAJE

      WRITE(6,100)MENSAJE
  100 FORMAT('ERROR :',A)

      STOP
      END
```

Solución 7.e

La solución del sistema por el método iterativo de Gauss-Seidel sería la siguiente:

$$\omega_0^{k+1} = (u_0 - 2\,k_1\,\omega_1^k)/(4\,k_1)$$

$$\omega_1^{k+1} = (u_1 - 2\,k_2\,\omega_0^{k+1} - 2\,k_2\,\omega_2^k)/(4(k_1 + k_2))$$

$$\vdots$$

$$\omega_i^{k+1} = (u_i - 2\,k_i\,\omega_{i-1}^{k+1} - 2\,k_{i+1}\,\omega_{i+1}^k)/(4(k_i + k_{i+1})) \qquad i = 1, \ldots, n-1$$

$$\vdots$$

$$\omega_{n-1}^{k+1} = (u_{n-1} - 2\,k_{n-1}\,\omega_{n-2}^{k+1} - 2\,k_n\,\omega_n^k)/(4(k_{n-1} + k_n))$$

$$\omega_n^k = (u_n - 2\,k_n\,\omega_{n-1}^{k+1})/(4\,k_n)$$

El método funcionará con total seguridad, pues la matriz $\boldsymbol{K}$ es diagonalmente dominante:

$$\begin{cases} |4\,k_1| > |2\,k_1| \\ |4(k_e + k_{e+1})| > |2\,k_e| + |2\,k_{e+1}| \\ |4\,k_n| > |2\,k_n| \end{cases}$$

Desde un punto de vista estructural, el método puede interpretarse de la siguiente forma:

- Se suponen unos giros inicialmente.
- Se repite hasta convergencia:
  - Se empotran todos los apoyos con sus giros actuales.
  - Se libera el nodo 0 y se recalcula su giro.

- Se vuelven a empotrar todos los apoyos.
- Se libera el nodo 1 y se recalcula su giro.
- Se procede igualmente para todos los nodos restantes.

Esta es la base de los antiguos métodos de cálculo estructural previos al análisis matricial, como los métodos de Cross, Kani, etc.

8.— Sea $\boldsymbol{A}$ una matriz simétrica y regular de orden n. Se desea encontrar la solución del sistema lineal de ecuaciones:

$$\boldsymbol{A}\,\boldsymbol{x} = \boldsymbol{b}$$

dado $\boldsymbol{b}$, se pide:

a) Escribir el algoritmo de Gauss sin pivotamiento **evitando las operaciones inútiles**. Calcular el número de operaciones necesarias para resolver el problema.

b) ¿Puede emplearse el algoritmo de Gauss con pivotamiento conservando la simetría? ¿Por qué?

Sugerencia: Recuérdese que en cada paso del proceso de eliminación, cuando se anulan los términos de la columna k-ésima por debajo del pivote, a_{kk}, sólo se recalcula la submatriz:

$$\begin{pmatrix} a_{k+1,k+1} & \cdots & a_{k+1,n} \\ \vdots & \ddots & \vdots \\ a_{n,k+1} & \cdots & a_{n,n} \end{pmatrix}$$

Solución 8.a

Partiendo del sistema de ecuaciones:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{Bmatrix}$$

tras finalizar el primero paso de normalización de la primera ecuación y eliminación en las restantes, el sistema tiene la forma siguiente:

$$\begin{bmatrix} a_{11} & a_{12}/a_{11} & a_{13}/a_{11} & \cdots & a_{1n}/a_{11} \\ 0 & a_{22} - a_{21}\frac{a_{12}}{a_{11}} & a_{23} - a_{21}\frac{a_{13}}{a_{11}} & \cdots & a_{2n} - a_{21}\frac{a_{1n}}{a_{11}} \\ 0 & a_{32} - a_{31}\frac{a_{12}}{a_{11}} & a_{33} - a_{31}\frac{a_{13}}{a_{11}} & \cdots & a_{3n} - a_{31}\frac{a_{1n}}{a_{11}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2} - a_{n1}\frac{a_{12}}{a_{11}} & a_{n3} - a_{n1}\frac{a_{13}}{a_{11}} & \cdots & a_{nn} - a_{n1}\frac{a_{1n}}{a_{11}} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{Bmatrix} = \begin{Bmatrix} b_1/a_{11} \\ b_2 - a_{21}\frac{b_1}{a_{11}} \\ b_3 - a_{31}\frac{b_1}{a_{11}} \\ \vdots \\ b_n - a_{n1}\frac{b_1}{a_{11}} \end{Bmatrix}$$

Se puede observar fácilmente que si la matriz era inicialmente simétrica, la submatriz con la que se trabajára en la fase siguiente también lo es. Las submatrices con las que se opera en cada paso conservan la simetría a lo largo de todo el proceso.

Por tanto podemos modificar el método de Gauss para evitar almacenar elementos fuera de la diagonal dos veces. Si lo planteamos almacenando la parte triangular superior de la matriz:

Triangularización /Eliminación
$$\left\{\begin{array}{l}\text{do } i = 1, n-1\\ \quad \text{do } k = i+1, n\\ \quad\quad c = a_{ik}/a_i i\\ \quad\quad a_{kj} = a_{kj} - c\, a_{ij} \quad ; \quad j = k, \ldots, n\\ \quad \text{enddo}\\ \text{enddo}\end{array}\right.$$

Elim. Térm. Indep.
$$\left\{\begin{array}{l}\text{do } i = 1, n-1\\ \quad \text{do } k = i+1, n\\ \quad\quad c = a_{ik}/a_i i\\ \quad\quad v_k = b_k - c\, b_i \quad ; \quad j = k, \ldots, n\\ \quad \text{enddo}\\ \text{enddo}\end{array}\right.$$

Resolución
$$\left\{\begin{array}{l} x_n = b_n/a_{nn}\\ x_i = \left(b_i - \sum_{j=i+1}^{n} a_{ij}\, x_j\right)/a_{ii}; \quad i = n-1, \ldots, 1 \quad \text{(Hacia atrás)}\end{array}\right.$$

Solución 8.b

No podemos plantear un algoritmo de Gauss con pivotamiento para matrices simétricas porque el pivotamiento destruye la simetría de la matriz.

El método de Gauss es conceptualmente más complicado que el método de Cholesky y no tiene ninguna ventaja por lo que normalmente no se utiliza. En general se prefiere utilizar el método de Cholesky para sistemas con matrices simétricas definidas positivas.