DOI: 10.22199/S07160917.1983.0006.00002

REVISTA PROYECCIONES Nº 6: 11-50.
DICIEMBRE 1983.

SOLUCIÓN NUMÉRICA DE ECUACIONES ELÍPTICAS POR EL MÉTODO DEL PRODUCTO TENSORIAL CON USO DE LA TRANSFORMADA RÁPIDA DE FOURIER

OSCAR ROJO J.*
RICARDO SOTO M.*

RESUMEN.

Se describe el método del Producto Tensorial para la solución numérica de Ecuaciones Diferenciales Parciales Elípticas Lineales Separables presentado originalmente en [3]. La solución por este método requiere del cómputo sucesivo de cuatro matrices. En este trabajo, tal cómputo es efectuado usando un algoritmo presentado en [5] que hace uso de la transformada rápida de Fourier. El método es aplicado a las ecuaciones de Poisson, de Helmholtz y Biarmónica con condiciones de Dirichlet en la frontera. Se dan ejemplos y se presentan los respectivos programas computacionales.

---

* Profesor Departamento de Matemáticas, Facultad de Ciencias, Universidad del Norte.

1. INTRODUCCION.

El Método del Producto Tensorial se basa en una idea simple y fundamental, que constituye la base del clásico método de separación de variables del análisis. Si el problema es separable, entonces la solución puede expresarse en términos de productos tensoriales de soluciones de problemas de menor dimensión y por lo tanto más simples. En el caso de las ecuaciones diferenciales parciales, esta idea implica que las matrices involucradas pueden ser expresadas en términos de productos tensoriales de matrices de menor orden.

En los casos más simples, la discretización de la ecuación diferencial conduce a una ecuación matricial de la forma

$$(I \otimes A + B \otimes I)v = g \tag{1.1}$$

donde A, B e I son matrices y v y g vectores.

Lo interesante es que el método del Producto Tensorial dá la solución exacta (salvo errores de redondeo) de la ecuación (1.1), la que puede ser computada con menos operaciones que las requeridas por el común de los esquemas iterativos de sobrerrelajación.

Sin embargo, nuestro propósito es aplicar al proceso de solución numérica de ecuaciones elípticas, el método del Producto Tensorial combinado con el algoritmo de la Transformada Rápida de Fourier (algoritmo FFT). Esta combinación dá soluciones directas que son computadas más eficientemente que por el método del Producto Tensorial.

Empezaremos por definir el producto tensorial de matrices y enunciar algunas de sus propiedades.

Definición 1.1.

Sean A y B matrices de orden mxn y kxl respectivamente con elementos a_{ij} y b_{ij}. El producto tensorial de A y B, denotado por $A \otimes B$ es la matriz de orden mk x nl

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ a_{31}B & a_{32}B & \cdots & a_{3n}B \\ \vdots & \vdots & & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{bmatrix}$$

Si n = l = 1, A y B son vectores columnas y su producto tensorial es un vector columna de mk componentes.

Propiedades:

1.2. $(A+B) \otimes C = A \otimes C + B \otimes C$

1.3. $A \otimes (B+C) = A \otimes B + A \otimes C$

1.4. $(A \otimes B)\ (C \otimes D) = AC \otimes BD$

1.5. $(A \otimes B)^T = A^T \otimes B^T$

1.6. $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$

1.7. Si x e y son autovectores de A y B, con autovalores λ y μ respectivamente, entonces $x \otimes y$ es un autovector de $A \otimes B$ con autovalor $\lambda\mu$.

2. EL METODO DEL PRODUCTO TENSORIAL.

Consideremos la ecuación diferencial parcial elíptica lineal de 2° orden en dos variables independientes x e y

(2.1) $L[u] = f$

donde el operador diferencial L tiene la forma $L = L_x + L_y$, con

(2.2) $$L_x = -a_2(x)\frac{\partial^2}{\partial x^2} + a_1(x)\frac{\partial}{\partial x} + a_0(x),\ a_2 > 0,\ a_0, a_1 \geq 0$$

(2.3) $$L_y = -b_2(x)\frac{\partial^2}{\partial y^2} + b_1(y)\frac{\partial}{\partial y} + b_0(y),\ b_2 > 0,\ a_0, a_1 \geq 0$$

o bien la ecuación

(2.4) $$-a_2(x)u_{xx}+a_1(x)u_x-b_2(y)u_{yy}+b_1(y)u_y+(a_0(x)+b_0(y))u = f(x,y)$$

sobre el rectángulo $R = \{(x,y)/a \leq x \leq b,\ c \leq y \leq d\}$.
Por simplicidad supondremos condiciones de borde del tipo Dirichlet.

Particionamos el dominio R colocando una malla definida por

(2.5) $$P_{h,k} = \{(x_i,y_j)/x_i=x_0+ih,\ y_j=y_0+jk,\ i=0, 1, \ldots n;\ j=0,1,\ldots, m\}$$

donde

$$x_0 = a,\ y_0 = c\ ,\ h = \frac{b-a}{n}\ ,\ k = \frac{d-c}{m}\ .$$

Sustituimos las derivadas en (2.4) por las aproximaciones por diferencias finitas

(2.6) $$u_{xx}(x_i,y_j) = \frac{u_{i-1,j} - 2u_{ij} + u_{i+1j}}{h^2} + O(h^2)$$

(2.7) $$u_x(x_i,y_j) = \frac{u_{i+1,j} - u_{i-1,j}}{2h} + O(h^2)$$ y

análogamente para $u_{yy}(x_i,y_j)$ y $u_y(x_i,y_j)$. Esto conduce a la ecuación de diferencias

$$(2.8)\quad \frac{1}{2h^2}\left[-(a_1h+2a_2)v_{i-1,j}+2(a_0h^2+2a_2)v_{ij}-(2a_2-a_1h)v_{i+1,j}\right]+$$

$$+\frac{1}{2k^2}\left[-(b_1k+2b_2)v_{i,j-1}+2(b_0k^2+2b_2)v_{ij}-(2b_2-b_1k)v_{i,j+1}\right]=f_{ij},$$

donde $v_{ij}=v(x_i,y_j)$ es una aproximación a $u(x_i,y_j)$ y las funciones a_k y b_k, $k=0,1,2$ son evaluadas en los puntos x_i e y_j respectivamente.

Aplicando (2.8) en cada punto interior (x_i,y_j) de la malla, $i=1,2,\ldots,n-1$; $j=1,2,\ldots,m-1$, se obtiene un sistema lineal de ecuaciones de tamaño $(n-1)(m-1)\times(n-1)(m-1)$ el que puede ser escrito en la forma

$$(2.9)\quad (I\otimes A+B\otimes I)v=g, \text{ donde}$$

$$(2.10)\quad A=\frac{1}{2h^2}\begin{bmatrix} 2(a_0h^2+2a_2) & -(2a_2-a_1h) & & & \\ -(a_1h+2a_2) & 2(a_0h^2+2a_2) & -(2a_2-a_1h) & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -(2a_2-a_1h) \\ & & & -(a_1h+2a_2) & 2(a_0h^2+2a_2) \end{bmatrix}$$

$$(2.11)\quad B=\frac{1}{2k^2}\begin{bmatrix} 2(b_0k^2+2b_2) & -(2b_2-b_1k) & & & \\ -(b_1k+2b_2) & 2(b_0k^2+2b_2) & -(2b_2-b_1k) & & \\ & \ddots & \ddots & \ddots & \\ & & & & -(2b_2-b_1k) \\ & & & -(b_1k+2b_2) & 2(b_0k^2+2b_2) \end{bmatrix}$$

Las matrices A y B son tridiagonales y de tamaño $(n-1)x(n-1)$ y $(m-1)x(m-1)$ respectivamente, mientras que los vectores v y g son de dimensión $(n-1)(m-1)$ y g incluye los valores de f y los de u en la frontera. Las matrices I, son matrices identidad de tamaño apropiado.

Para h y k suficientemente pequeños, las matrices A y B resultan ser del tipo diagonal dominante. En efecto eligiendo h tal que $a_1h-2a_2 < 0$ se tiene $|a_1h\pm 2a_2| = 2a_2 \pm a_1h$ y $|a_1h + 2a_2| + |a_1h - 2a_2| = 4a_2$ que es menor o igual que el correspondiente elemento de lo diagonal $4a_2 + 2a_0h^2$ pues $a_0 \geq 0$. El mismo argumento es válido para B con k suficientemente pequeño.

La matriz $(I \otimes A + B \otimes I)$ tiene a lo más cinco elementos no nulos sobre cada fila y del carácter diagonal dominante que tienen las matrices A y B con elementos positivos en la diagonal, se sigue que $(I \otimes A + B \otimes I)$ es también una matriz diagonal dominante con desigualdad estricta para al menos una fila. Puede probarse también que $(I \otimes A + B \otimes I)$ es irreducible y luego por lo anterior irreduciblemente diagonal dominante. Entonces a partir del Teorema de Gerschgorin se sigue que $(I \otimes A + B \otimes I)$ es una matriz no singular. Así el sistema (2.9) tiene una única solución.

Si las matrices A y B son diagonalizables, entonces existen matrices P y Q no singulares tales que

$$(2.12) \quad P^{-1}AP = \Lambda(A) \qquad \text{y} \qquad Q^{-1}BQ = \Lambda(B),$$

donde $\Lambda(A)$ y $\Lambda(B)$ son matrices diagonales cuyos elementos diagonales son los autovalores de A y B respectivamente.

Aplicando propiedades del producto tensorial se tiene que

$$Q^{-1} \otimes P^{-1} \; (I \otimes A + B \otimes I) \; Q \otimes P =$$

$$= I \otimes P^{-1} AP + Q^{-1} BQ \otimes I$$

$$= I \otimes \Lambda(A) + \Lambda(B) \otimes I,$$

de donde:

$$(2.13) \quad (I \otimes A + B \otimes I)^{-1} = Q \otimes P (I \otimes \Lambda(A) + \Lambda(B) \otimes I)^{-1} Q^{-1} \otimes P^{-1}$$

Así, la solución del sistema matricial (2.9) queda dada por

$$(2.14) \quad v = Q \otimes P \; (I \otimes \Lambda(A) + \Lambda(B) \otimes I)^{-1} Q^{-1} \otimes P^{-1} g, \text{ donde}$$

la matriz $(I \otimes \Lambda(A) + \Lambda(B) \otimes I)$ es diagonal y su inversa se computa trivialmente. La solución dada por (2.14) es exacta, salvo errores de redondeo.

En muchos casos de interés práctico, los autovectores y autovalores de las matrices A y B son conocidos o bien pueden ser obtenidos con relativa facilidad dado que A y B son tridiagonales.

En orden a computar la solución v (no es recomendable computar la inversa (2.13) para luego multiplicarla por g), podemos escribir (2.14) para cada punto interior (x_i, y_j) de la malla en términos de elementos matriciales como

$$(2.15)\quad v_{ij} = \sum_{\beta=1}^{m-1} q_{i\beta} \sum_{\alpha=1}^{n-1} p_{j\alpha} \frac{1}{\lambda_\alpha(A)+\lambda_\beta(B)} \sum_{k=1}^{m-1} q'_{\beta k} \sum_{l=1}^{n-1} p'_{\alpha l}\, g_{kl}\ ,$$

donde $P = (p_{j\alpha})$, $P^{-1} = (p'_{\alpha l})$, $Q = (q_{i\beta})$, $Q^{-1} = (q'_{\beta k})$ y

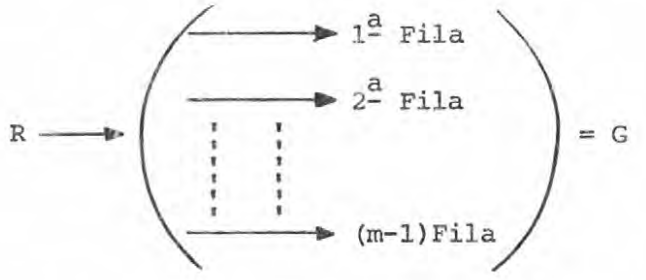
$\lambda_\alpha(A)$ y $\lambda_\beta(B)$ son los autovalores de A y B respectivamente.

La solución v puede ser obtenida computando sucesivamente tres matrices intermedias R, S y T y la matriz solución V según el siguiente alforitmo:

1º) Computar la matriz

$$(2.16)\quad R = (r_{k\alpha}) = \sum_{l=1}^{n-1} p'_{\alpha l}\, g_{kl}$$

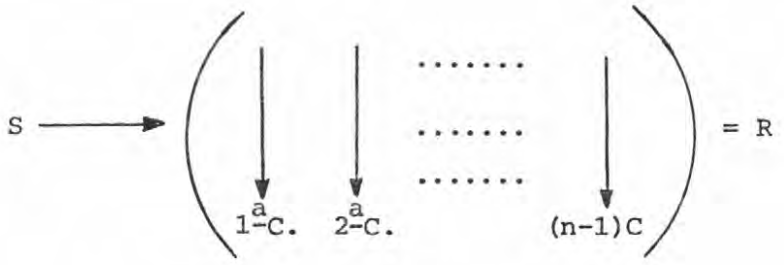
La k-ésima fila de R se forma multiplicando (en el sentido del producto escalar de vectores) cada fila de P^{-1} por la k-ésima fila de G, donde G es la "matriz del segundo miembro" formada haciendo una partición adecuada del vector g de constantes del segundo miembro de (2.9). La k-ésima fila de R así computada, es almacenada en el espacio correspondiente a la k-ésima fila de G, de modo que R ocupará finalmente el espacio que originalmente ocupaba G.



2°) Computar la matriz

$$(2.17) \qquad S = (S_{\beta\alpha} = \frac{1}{\lambda_\alpha(A) + \lambda_\beta(B)} \sum_{k=1}^{m-1} q'_{\beta k}\, r_{k\alpha})$$

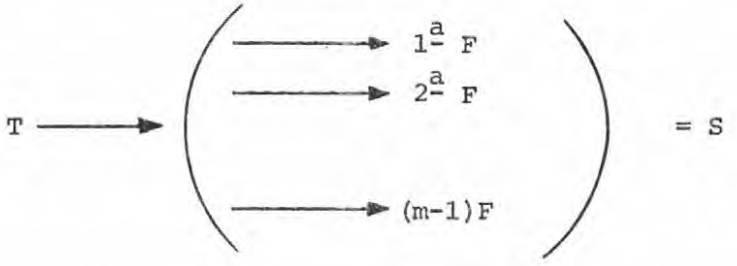
La α-ésima columna de S es obtenida multiplicando cada fila de Q^{-1} por la α-ésima columna de R. S es almacenada en R por columnas



3°) Computar la matriz

$$(2.18) \qquad T = (t_{\alpha j} = \sum_{\alpha=1}^{n-1} p_{j\alpha}\, s_{\beta\alpha})$$

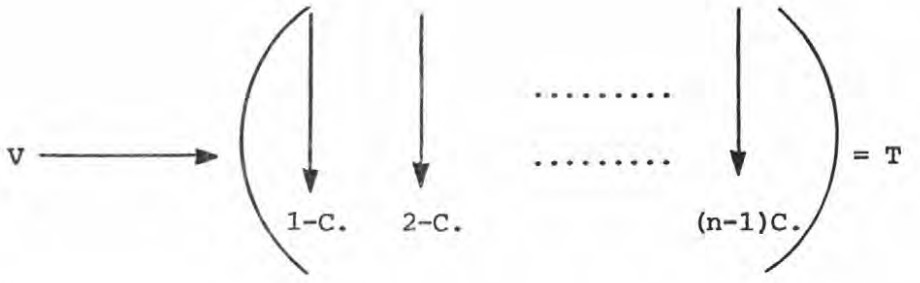
La β-ésima fila de T es computada multiplicando cada fila de P por la β-ésima fila de S. T es almacenada en S por filas



4°) Computar la matriz

$$(2.19) \qquad V = (v_{ij} = \sum_{\beta=1}^{m-1} q_{i\beta}\, t_{\beta j})$$

La j-ésima columna de V es computada multiplicando cada fila de Q por la j-ésima columna de T. V es almacenada en T por columnas



3. PRODUCTO TENSORIAL-TRANSFORMADA RAPIDA DE FOURIER (TP-FFT).

El cómputo de las matrices R, S, T y V de la sección 2 puede ser efectuado muy eficientemente por medio del algoritmo de la Transformada Rápida de Fourier (FFT). En particular queremos aplicar al cómputo de dichas matrices el algoritmo presentado en [5] para computar el producto Qc, donde Q es una matriz y c un vector. Este algoritmo hace uso de la FFT y puede ser aplicado en combinación con el método del Productor Tensorial (TP) en la solución de diversos problemas de interés práctico. Esto puede hacerse directamente debido a que las matrices P y Q involucradas en el cómputo de (2.15) en estos problemas, coinciden con la matriz Q del producto Qc mencionado.

3.1. La Ecuación de Poisson.

Consideremos la ecuación de Poisson

$$(3.1) \qquad -\Delta u = -(u_{xx} + u_{yy}) = f(x,y)$$

con condiciones de borde de Dirichlet sobre el rectángulo

$$R = \{(x,y)/a \leq x \leq b,\ c \leq y \leq d\}.$$

Particionamos el dominio R como en (2.5) con un espaciado horizontal $h = \frac{b-a}{n}$ y un espaciado vertical $k = \frac{d-c}{m}$.

Aproximando el Laplaciano por aproximaciones por diferencias finitas obtenemos la ecuación de diferencias

$$(3.2)\qquad \frac{1}{h^2}\,(-v_{i-1,j}+2v_{ij}-v_{i+1,j}) + \frac{1}{k^2}\,(-v_{i,j-1}+2v_{ij}-v_{i,j+1}) = f_{ij}$$

que conduce a la ecuación matricial

$$(3.3)\qquad (I \otimes A + B \otimes I)v = g$$

cuya solución como hemos visto, viene dada por (2.14) y (2.15) y donde

$$(3.4)\qquad A = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & & & \bigcirc \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & \ddots & \ddots & \ddots \\ & & & -1 & 2 \\ \bigcirc & & & & \end{bmatrix} \quad \text{y} \quad B = \frac{h^2}{k^2} A$$

El vector g incluye los valores de f y las condiciones de borde. A es de tamaño (n-1)x(n-1) y B de tamaño (m-1)x(m-1). Respecto de A y B (que sólo se diferencian por su tamaño y los factores $\frac{1}{h^2}$ y $\frac{1}{k^2}$) se tiene bastante información. Son conocidos sus autovectores y autovalores y las matrices que las diagonalizan, las que además son ortogonales.

Los autovalores de A en (3.4) están dados por:

$$(3.5)\qquad \lambda_\alpha = \frac{1}{h^2}\,(2-2\cos\frac{\alpha\pi}{n}),\ \alpha = 1, 2, \ldots, n-1$$

y sus autovectores tienen componentes

$$(3.6)\qquad x_i^{(\alpha)} = \operatorname{sen}\frac{i\alpha\pi}{n}\ ,\ \text{con norma}\ \sqrt{\frac{n}{2}}\ ,\ i = 1, 2, \ldots, n-1.$$

Por lo tanto, la matriz P tal que $P^{-1}AP = \Lambda(A) = \operatorname{diag}\{\lambda_\alpha\}$ es

$$(3.7)\qquad P = \sqrt{\frac{2}{n}}\begin{bmatrix} \operatorname{sen}\frac{\pi}{n} & \operatorname{sen}\frac{2\pi}{n} & \cdots\cdots & \operatorname{sen}\frac{(n-1)\pi}{n} \\ \operatorname{sen}\frac{2\pi}{n} & \operatorname{sen}\frac{4\pi}{n} & \cdots\cdots & \operatorname{sen}\frac{2(n-1)\pi}{n} \\ \vdots & \vdots & & \vdots \\ \operatorname{sen}\frac{(n-1)\pi}{n} & \operatorname{sen}\frac{2(n-1)\pi}{n} & \cdots\cdots & \operatorname{sen}\frac{(n-1)^2\pi}{n} \end{bmatrix}$$

Los autovalores y autovectores de B son respectivamente

$$(3.8)\qquad \lambda_\beta = \frac{1}{k^2}(2-2\cos\frac{\beta\pi}{m}) \text{ y } x_j^{(\beta)} = \operatorname{sen}\frac{j\beta\pi}{m}, \text{ con norma } \sqrt{\frac{m}{2}},$$

donde $\beta = 1, 2, \ldots, m-1$ y $j = 1, 2, \ldots, m-1$. La matriz que diagonaliza a B es

$$(3.9)\qquad Q = (q_{ij}) = \sqrt{\frac{2}{m}}\,(\operatorname{sen}\frac{ij\pi}{m}) \text{ de dimensión } (m-1)\times(m-1).$$

Las matrices P y Q son ortogonales, simétricas, y de idéntica forma y son además coincidentes con la matriz Q del algoritmo para computar el producto Qc ya mencionado. Es posible entonces, aplicar directamente este algoritmo al cómputo de las matrices R, S, T y V definidas por (2.16) - (2.19), obteniendo así una solución directa al problema (3.1).

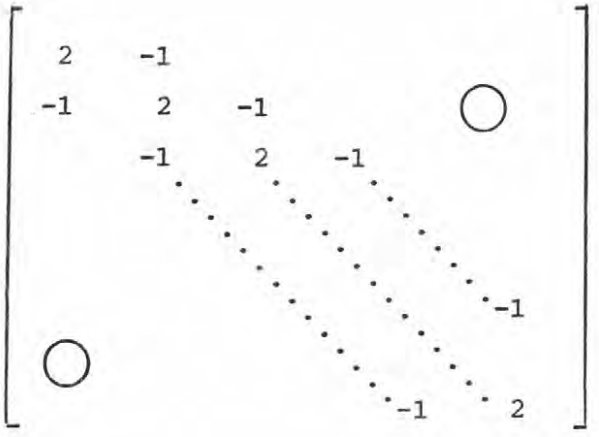
Observemos que si R es un cuadrado particionado uniformemente (h=k), entonces (3.2) pasa a ser

$$(3.10)\qquad (-v_{i-1,j}+2v_{ij}-v_{i+1,j}) + (-v_{i,j-1}+2v_{ij}-v_{i,j+1}) = h^2 f_{ij}$$

y (3.3) queda

$$(3.11)\qquad (I \otimes A + A \otimes I)v = g, \text{ donde}$$

(3.12) $A =$



con autovalores $\lambda_\alpha = 2-2\cos\frac{\alpha\pi}{n}$. Las matrices P y Q son tales que

$P = Q$ y $P^{-1} = P$. Entonces la relación (2.15) toma la forma

$$(3.13)\quad v_{ij} = \sum_{\beta=1}^{n-1} p_{i\beta} \sum_{\alpha=1}^{n-1} p_{j\alpha} \frac{1}{\lambda_\alpha + \lambda_\beta} \sum_{k=1}^{n-1} p_{\beta k} \sum_{l=1}^{n-1} p_{\alpha l}\, g_{kl}$$

3.2. Algoritmo para el cómputo del producto Qc.

Aplicamos aquí la Transformada Rápida de Fourier para computar el producto $d = Qc$, donde $c = [c_1, c_2, \ldots, c_{n-1}]^T$ es un vector de componentes reales con $n=2^{\gamma+1}$ y Q la matriz ortogonal

$$Q = (q_{ij}) = \sqrt{\frac{2}{n}}\ (\operatorname{sen}\frac{ij\pi}{n}),\ \text{es decir,}$$

$Q = P$ es la matriz definida en (3.7).

Sea

$$(3.14)\quad d_k = \sqrt{\frac{2}{n}} \sum_{j=0}^{n-1} c_j \operatorname{sen}\frac{kj\pi}{n} \text{ con } c_0 = 0,\ \text{la k-ésima componente}$$

del vector Qc.

Definimos para $k = 0, 1, \ldots N-1$, $N = \frac{n}{2}$:

$$p_k = \sqrt{\frac{n}{2}}\, d_{2k} \quad y$$

$$q_k = \sqrt{\frac{n}{2}}\, d_{2k+1}. \quad \text{Entonces se tiene}$$

$$(3.15) \qquad p_k = \sum_{j=0}^{n-1} c_j \operatorname{sen} \frac{2kj\pi}{n}$$

$$(3.16) \qquad q_k = \sum_{j=0}^{n-1} c_j \operatorname{sen} \frac{(2k+1)j\pi}{n}$$

Sean

$$(3.17) \qquad C(k) = \sum_{j=0}^{n-1} c_j\, e^{\frac{-ikj(2\pi)}{n}} \qquad y$$

$$(3.18) \qquad \tilde{C}(k) = \sum_{j=0}^{n-1} \tilde{c}_j\, e^{\frac{-ikj(2\pi)}{n}} \qquad \text{donde}$$

$$(3.19) \qquad \tilde{c}_j = c_j\, e^{\frac{-ij\pi}{n}}, \; i = \sqrt{-1}, \; j = 1, 2, \ldots, n-1, \; \tilde{c}_0 = 0$$

Entonces podemos escribir

$$p_k = -\operatorname{Im} C(k) \qquad y \qquad q_k = -\operatorname{Im} \tilde{C}(k).$$

Para computar $C(k)$ separamos la sucesión c_j en dos nuevas sucesiones como sigue:

(3.20) $h_j = c_{2j}$; $g_j = c_{2j+1}$, $j = 0, 1, \ldots, N-1$, $h_0 = 0$

Computamos simultáneamente las transformadas discretas

(3.21) $H(k) = \sum_{j=0}^{N-1} h_j \, e^{\frac{-ikj(2\pi)}{n}}$ de h_j y

(3.22) $G(k) = \sum_{j=0}^{N-1} g_j \, e^{\frac{-ikj(2\pi)}{n}}$ de g_j, aplicando el algoritmo FFT

a la suma

(3.23) $y_j = h_j + ig_j$, $j = 0, 1, \ldots, N-1$. Obtenemos

(3.24) $Y(k) = R(k) + iI(k)$, donde $R(k)$ e $I(k)$ son las partes real e imaginaria de $Y(k)$ respectivamente.

Puede probarse que $H(k) = \frac{R(k)+R(N-k)}{2} + i\,\frac{I(k)-I(N-k)}{2} = R_e + iI_o$ y

$$G(k) = \frac{I(k)+I(N-k)}{2} - i\,\frac{R(k)-R(N-k)}{2} = I_e - iR_o$$

donde los subíndices e y o indican respectivamente las partes par e impar de R e I.

Escribiendo $C(k) = H(k) + e^{\frac{-ik\pi}{N}} G(k)$ se tiene entonces

(3.25) $C(k) = [R_e(k) + I_e(k) \cos\frac{k\pi}{N} - R_o \operatorname{sen}\frac{k\pi}{N}] +$

$+ i\,[I_o(k) - I_e(k) \operatorname{sen}\frac{k\pi}{N} - R_o(k) \cos\frac{k\pi}{N}]$ y de aquí

(3.26) $d_{2k} = \sqrt{\frac{2}{n}} \left[R_o(k) \cos \frac{k\pi}{N} + I_e(k) \operatorname{sen} \frac{k\pi}{N} - I_o(k)\right]$, $k = 1, 2, \ldots, N-1$.

Repitiendo el argumento anterior, para computar $\tilde{C}(k)$ hacemos

(3.27) $\tilde{h}_j = \tilde{c}_{2j}$; $\tilde{g}_j = \tilde{c}_{2j+1}$, $j = 0, 1, \ldots N-1$ $\tilde{h}_0 = 0$ y

computamos sucesivamente las transformadas discretas $\tilde{H}(k)=R_1(k)+iI_1(k)$ y $\tilde{G}(k)=R_2(k)+iI_2(k)$ de las sucesiones complejas $\tilde{h}_j$ y $\tilde{g}_j$, donde R_1 e I_1 representan las partes real e imaginaria de $\tilde{H}(k)$ respectivamente, mientras que R_2 e I_2 son las partes real e imaginaria de $\tilde{G}(k)$.

Escribiendo $\tilde{C}(k) = \tilde{H}(k) + e^{\frac{-ik\pi}{N}} \tilde{G}(k)$ se obtiene finalmente

(3.28) $d_{2k+1} = \sqrt{\frac{2}{n}} \left[R_2(k) \operatorname{sen} \frac{k\pi}{N} - I_1(k) - I_2(k) \cos \frac{k\pi}{N}\right]$

Resumiendo, el algoritmo queda:

1°) Computar c_j de (3.19).

2°) Separar las sucesiones c_j y $\tilde{c}_j$ según (3.20) y (3.27).

3°) Computar mediante el algoritmo FFT las transformadas discretas de $y_j = h_j + ig_j$, $\tilde{h}_j$ y $\tilde{g}_j$.

4°) Finalmente computar d_{2k} y d_{2k+1} según (3.26) y (3.28) respectivamente.

3.3. La Ecuación de Helmholtz.

Consideremos la ecuación de Helmholtz

(3.29) $-\Delta u + \sigma u = f(x,y)$, $\sigma \geq 0$

con condiciones de borde de Dirichlet sobre el rectángulo [a,b]x[c,d]. Asumimos una partición del dominio con un espaciado horizontal h y un espaciado vertical k. Aproximando el operador Laplaciano como antes obtenemos la ecuación en diferencias:

$$(3.30)\quad \frac{1}{h^2}(-v_{i-1,j}+2v_{ij}-v_{i+1,j}) + \frac{1}{k^2}(-v_{i,j-1}+2v_{ij}-v_{i,j+1}) + \sigma v_{ij} = f_{ij}$$

que puede escribirse como una ecuación matricial en notación tensorial como

$$(3.31)\quad (I \otimes A + B \otimes I + \sigma\, I \otimes I)v = g$$

Al igual que antes, las matrices A y B corresponden a aplicar la aproximación de diferencias finitas en sentido horizontal y vertical respectivamente y son las mismas definidas en (3.4). El vector g incluye los valores de u en la frontera y los valores de f.

La ecuación (3.31) puede escribirse también en la forma

$$(3.32)\quad (I \otimes (A+\sigma I) + B \otimes I)v = g$$

es decir en la forma (3.3) con $A+\sigma I$ en lugar de A.

Observemos que para $\sigma > 0$, $(I \otimes (A+\sigma I) + B \otimes I)$ es una matriz diagonal dominante estricta (para h y k suficientemente pequeños).

Ahora bien, A+σI tiene autovalores dados por

$$(3.33) \quad \lambda_j = \frac{1}{h^2}\left(2-2\cos\frac{j\pi}{n}\right) + \sigma \ , \ j = 1, 2, \ldots, n-1 \quad \text{y}$$

autovectores con norma $\sqrt{\frac{n}{2}}$ definidos por

$$(3.34) \quad x_i^{(j)} = \operatorname{sen}\frac{ij\pi}{n} \ , \ i = 1, 2, \ldots n-1$$

Es decir, la matriz P que diagonaliza a (A+σI) es la misma que diagonaliza a A. Luego, para hallar una solución de la ecuación de Helmholtz (3.29) se aplica el mismo procedimiento empleado para el problema (3.1) de Poisson, es decir computar las matrices R, S, T y V por medio del algoritmo para computar Qc.

Si el dominio R es un cuadrado particionado uniformemente (h=k) se obtiene la ecuación en diferencias

$$(3.35) \quad (-v_{i-1,j}+2v_{ij}-v_{i+1,j})+(-v_{i,j-1}+2v_{ij}-v_{i,j+1})+\sigma h^2 v_{ij} = h^2 f_{ij}$$

que puede escribirse como

$$(3.36) \quad (I \otimes (A+\sigma h^2 I) + A \otimes I)\, v = g$$

donde A tiene autovalores $\lambda_\alpha = 2-2\cos\frac{\alpha\pi}{n}$, mientras que los autovalores de $(A+\sigma h^2 I)$ son $\lambda_j = 2+\sigma h^2-2\cos\frac{j\pi}{n}$.

Aunque sólo hemos establecido aquí condiciones de frontera del

tipo Dirichlet, el método puede adaptarse perfectamente bien a otro tipo de condiciones. Por ejemplo, si condiciones de frontera periódicas son dadas con las ecuaciones de Poisson y de Helmholtz, las matrices A y B son modificadas por la aparición de un -1 en las esquinas inferior izquierda y superior derecha. Estas matrices son del tipo llamado "matriz circulante" y sus autovalores y autovectores son conocidos.

3.3. La Ecuación Biarmónica.

Consideremos la ecuación biarmónica no homogénea

$$u_{xxxx} + 2u_{xxyy} + u_{yyyy} = f(x,y) \tag{3.37}$$

con condiciones de frontera dadas para u y sus segundas derivadas parciales u_{xx} y u_{yy} sobre el cuadrado $R = [a,b]x[a,b]$.

Claramente la ecuación (3.37) es equivalente a

$$-\Delta(-\Delta u) = f(x,y) \tag{3.38}$$

lo que nos permite resolver (3.37) resolviendo dos ecuaciones de Poisson:

$$1^\circ \text{ Resolver} \begin{cases} -\Delta v = f \\ v = g_1 \end{cases}, \text{ donde } -\Delta u = v$$

$$2^\circ \text{ Resolver} \begin{cases} -\Delta u = v \\ u = g_2 \end{cases}$$

En ambos problemas puede procederse del mismo modo que en el caso de la ecuación de Poisson, puesto que las matrices involucradas son las mismas.

Sin embargo es preferible resolver la ecuación (3.37) directamente del modo siguiente:

Aproximamos el operador diferencial de cuarto orden

$$\left(\frac{\partial}{\partial x^2} + \frac{\partial}{\partial y^2}\right)^2 = \frac{\partial^4}{\partial x^4} + 2\frac{\partial^4}{\partial x^2 \partial y^2} + \frac{\partial^4}{\partial y^4}$$

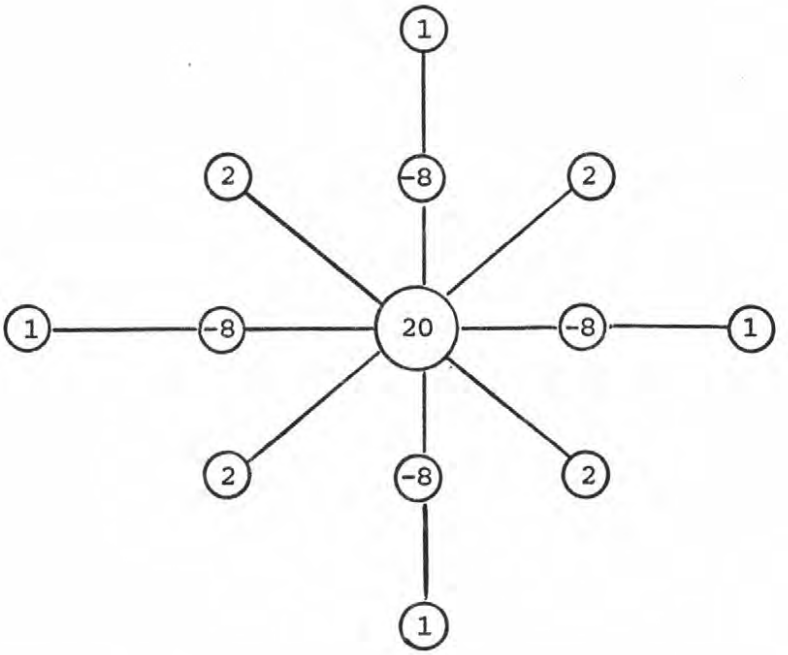
por una fórmula de 13 puntos obtenida aproximando dos veces el Laplaciano. Esta fórmula es

$$(3.39)\quad \Delta^2 u_{ij} = \frac{1}{h^4}\,[20u_{ij} - 8u_{i-1j} - 8u_{i+1,j} - 8u_{i,j-1} - 8u_{i,j+1} +$$

$$+ 2u_{i-1,j-1} + 2u_{i-1,j+1} + 2u_{i+1,j-1} + 2u_{i+1,j+1} +$$

$$+ u_{i-2,j} + u_{i+2,j} + u_{i,j-2} + u_{i,j+2}],$$

que puede representarse en la figura siguiente



Separar las aproximaciones en sentido horizontal, vertical y mixto conduce a la ecuación

$$(3.40) \quad (I \otimes A + A \otimes I)^2 v = (I \otimes A^2 + 2A \otimes A + A^2 \otimes I)v = g$$

Entonces,

$$(3.41) \quad v = P \otimes P \, [I \otimes \Lambda^2(A) + 2\Lambda(A) \otimes \Lambda(A) + \Lambda^2(A) \otimes I]^{-1} P^{-1} \otimes P^{-1} \, g,$$

con solución explícita dada por:

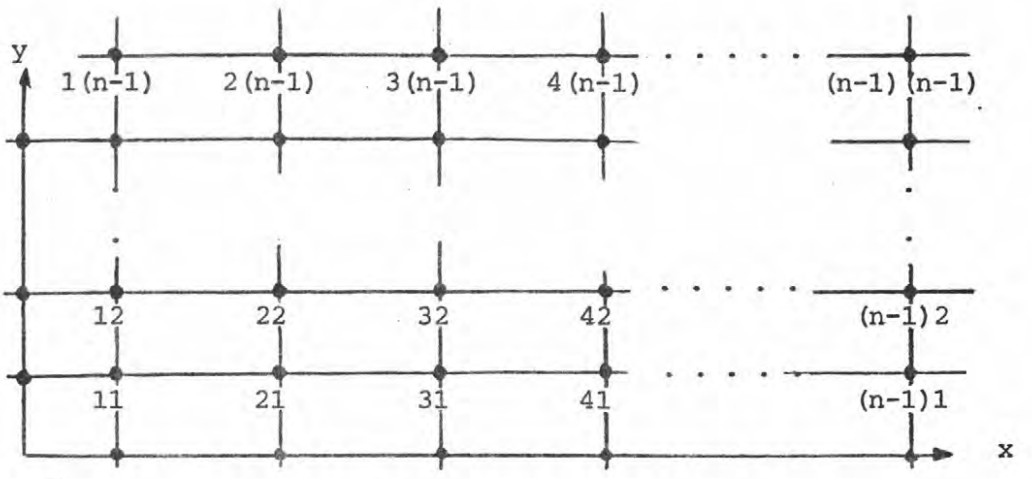
$$(3.42) \quad v_{ij} = \sum_{\beta=1}^{n-1} p_{i\beta} \sum_{\alpha=1}^{n-1} p_{j\alpha} \frac{1}{\lambda_\alpha^2 + 2\lambda_\alpha\lambda_\beta + \lambda_\beta^2} \sum_{k=1}^{n-1} p_{\beta k} \sum_{l=1}^{n-1} p_{\alpha l} \, g_{kl}$$

Nuevamente λ_α y λ_β son nuestros conocidos autovalores de A en (3.12), mientras que la matriz P que diagonaliza a A ha sido definida en (3.7). Así, podemos aplicar una vez más el algoritmo "Qc" para computar las matrices R, S, T y V que permiten obtener (3.42).

Queremos comentar aquí, que el método combinado Producto Tensorial Transformada Rápida de Fourier, requiere para computar la solución de los problemas presentados, almacenar en memoria solo una matriz de dimensión (n-1)x(m-1), que inicialmente corresponde a la matriz G del segundo miembro. Todas las siguientes matrices que se computan son almacenadas sucesivamente en G.

4. EJEMPLOS.

En los ejemplos que se presentan R = [0,1]x[0,1] es discretizado en una malla uniforme de paso $h = \frac{1}{n}$, la que es numerada de la forma siguiente



Los resultados numéricos están dados en una matriz de componentes v_{ij} de modo que el valor v_{ij} es una aproximación para $u\left(\frac{i}{n}, \frac{j}{n}\right)$.

Ejemplo 4.1.

$-\Delta u = \text{sen } x + \text{sen } y$

$u = \text{sen } x + \text{sen } y, \ (x,y) \in \partial R$

La solución exacta es $u(x,y) = \text{sen } x + \text{sen } y$. Con n=16 obtuvimos las siguientes aproximaciones para $u(x,y)$:

RESULTADOS FINALES

---------- ** ----------

0.1249196	0.1871358	0.2488651	0.3098664	0.3699015	0.4287360	0.4861401	0.5418896	0.5957669	0.6475615
0.6970712	0.7441026	0.7884719	0.8300059	0.8685424					
0.1871358	0.2493528	0.3110828	0.3720846	0.4321202	0.4909551	0.5483594	0.6041092	0.6579865	0.7097812
0.7592906	0.8063217	0.8506905	0.8922237	0.9307591					
0.2488651	0.3110828	0.3728133	0.4338157	0.4938517	0.5526869	0.6100915	0.6658414	0.7197188	0.7715134
0.8210226	0.8680533	0.9124217	0.9539542	0.9924887					
0.3098664	0.3720846	0.4338157	0.4948184	0.5548548	0.6136902	0.6710950	0.7268450	0.7807225	0.8325169
0.8820261	0.9290565	0.9734243	1.0149563	1.0534902					
0.3699015	0.4321202	0.4938517	0.5548548	0.6148914	0.6737271	0.7311321	0.7868821	0.8407596	0.8925540
0.9420630	0.9890931	1.0334606	1.0749922	1.1135254					
0.4287360	0.4909551	0.5526869	0.6136902	0.6737271	0.7325629	0.7899680	0.8457181	0.8995956	0.9513899
1.0008987	1.0479287	1.0922959	1.1338271	1.1723599					
0.4861401	0.5483594	0.6100915	0.6710950	0.7311321	0.7899680	0.8473732	0.9031233	0.9570008	1.0087951
1.0583038	1.1053336	1.1497006	1.1912316	1.2297640					
0.5418896	0.6041092	0.6658414	0.7268450	0.7868821	0.8457181	0.9031233	0.9588735	1.0127509	1.0645452
1.1140539	1.1610836	1.2054505	1.2469813	1.2855136					
0.5957669	0.6579865	0.7197188	0.7807225	0.8407596	0.8995956	0.9570008	1.0127509	1.0666283	1.1184226
1.1679313	1.2149610	1.2593279	1.3008586	1.3393908					
0.6475615	0.7097812	0.7715134	0.8325169	0.8925540	0.9513899	1.0087951	1.0645452	1.1184226	1.1702169
1.2197256	1.2667554	1.3111223	1.3526532	1.3911855					
0.6970712	0.7592906	0.8210226	0.8820261	0.9420630	1.0008987	1.0583038	1.1140539	1.1679313	1.2197256
1.2692344	1.3162643	1.3606315	1.4021625	1.4406950					
0.7441026	0.8063217	0.8680533	0.9290565	0.9890931	1.0479287	1.1053336	1.1610836	1.2149610	1.2667554
1.3162643	1.3632945	1.4076619	1.4491934	1.4877263					
0.7884719	0.8506905	0.9124217	0.9734243	1.0334606	1.0922959	1.1497006	1.2054505	1.2593279	1.3111223
1.3606315	1.4076619	1.4520298	1.4935618	1.5320954					
0.8300059	0.8922237	0.9539542	1.0149563	1.0749922	1.1338271	1.1912316	1.2469813	1.3008586	1.3526532
1.4021625	1.4491934	1.4935618	1.5350946	1.5736292					
0.8685424	0.9307591	0.9924887	1.0534902	1.1135254	1.1723599	1.2297640	1.2855136	1.3393908	1.3911855
1.4406950	1.4877263	1.5320954	1.5736292	1.6121652					

Ejemplo 4.2.

$$-\Delta u + 2u = 2xy$$

$$u = xy, \quad (x,y) \in \partial R$$

La solución exacta es u = xy. Con n=8 obtuvimos una solución numérica exacta. El error de discretización es cero.

RESULTADOS FINALES

---------- ** ----------

0.0156250	0.0312500	0.0468750	0.0625000	0.0781250	0.0937500	0.1093750
0.0312500	0.0625000	0.0937500	0.1250000	0.1562500	0.1875000	0.2187500
0.0468750	0.0937500	0.1406250	0.1875000	0.2343750	0.2812500	0.3281250
0.0625000	0.1250000	0.1875000	0.2500000	0.3125000	0.3750000	0.4375000
0.0781250	0.1562500	0.2343750	0.3125000	0.3906250	0.4687500	0.5468750
0.0937500	0.1875000	0.2812500	0.3750000	0.4687500	0.5625000	0.6562500
0.1093750	0.2187500	0.3281250	0.4375000	0.5468750	0.6562500	0.7656250

Ejemplo 4.3.

$$-\Delta u + \pi^2 u = 2\pi^2 (\cos \pi x + \cos \pi y)$$

$$u = \cos \pi x + \cos \pi y, \quad (x,y) \in \partial R$$

La solución es $u(x,y) = \cos \pi x + \cos \pi y$. Con n=16 obtuvimos los siguientes resultados:

RESULTADOS FINALES

---------- ** ----------

1.9618079	1.9050227	1.8126672	1.6883169	1.5367646	1.3638425	1.1762016	0.9810560	0.7859079	0.5982585
0.4253206	0.2737410	0.1493455	0.0569150	-0.0000001					
1.9050227	1.8483189	1.7560074	1.6316719	1.4801130	1.3071691	1.1194963	0.9243128	0.7291247	0.5414365
0.3684637	0.2168560	0.0924407	0.0000001	-0.0569150					
1.8126672	1.7560074	1.6637225	1.5393970	1.3878342	1.2148761	1.0271817	0.8319725	0.6367574	0.4490433
0.2760484	0.1244241	-0.0000001	-0.0924408	-0.1493458					
1.6883169	1.6316719	1.5393970	1.4150742	1.2635067	1.0905379	0.9028285	0.7076020	0.5123692	0.3246390
0.1516315	0.0000000	-0.1244242	-0.2168559	-0.2737411					
1.5367646	1.4801130	1.3878342	1.2635067	1.1119324	0.9389543	0.7512340	0.5559959	0.3607520	0.1730128
0.0000000	-0.1516315	-0.2760486	-0.3684637	-0.4253207					
1.3638425	1.3071691	1.2148761	1.0905379	0.9389543	0.7659675	0.5782388	0.3829930	0.1877429	0.0000000
-0.1730129	-0.3246390	-0.4490435	-0.5414364	-0.5982587					
1.1762016	1.1194963	1.0271817	0.9028285	0.7512340	0.5782388	0.3905036	0.1952530	0.0000001	-0.1877428
-0.3607519	-0.5123691	-0.6367574	-0.7291246	-0.7859080					
0.9810560	0.9243128	0.8319725	0.7076020	0.5559959	0.3829930	0.1952530	0.0000000	-0.1952528	-0.3829930
-0.5559958	-0.7076019	-0.8319725	-0.9243127	-0.9810561					
0.7859079	0.7291247	0.6367574	0.5123692	0.3607520	0.1877429	0.0000001	-0.1952528	-0.3905033	-0.5782386
-0.7512338	-0.9028283	-1.0271815	-1.1194960	-1.1762016					
0.5982585	0.5414365	0.4490433	0.3246390	0.1730128	0.0000000	-0.1877428	-0.3829930	-0.5782386	-0.7659675
-0.9389542	-1.0905378	-1.2148760	-1.3071690	-1.3638427					
0.4253206	0.3684637	0.2760484	0.1516315	0.0000000	-0.1730129	-0.3607519	-0.5559958	-0.7512338	-0.9389542
-1.1119322	-1.2635065	-1.3878341	-1.4801127	-1.5367646					
0.2737410	0.2168560	0.1244241	0.0000000	-0.1516315	-0.3246390	-0.5123691	-0.7076019	-0.9028283	-1.0905378
-1.2635065	-1.4150740	-1.5393969	-1.6316717	-1.6883169					
0.1493455	0.0924407	-0.0000001	-0.1244242	-0.2760486	-0.4490435	-0.6367574	-0.8319725	-1.0271815	-1.2148760
-1.3878341	-1.5393969	-1.6637224	-1.7560072	-1.8126673					
0.0569150	0.0000001	-0.0924408	-0.2168559	-0.3684637	-0.5414364	-0.7291246	-0.9243127	-1.1194960	-1.3071691
-1.4801127	-1.6316717	-1.7560072	-1.8483186	-1.9050227					
-0.0000001	-0.0569150	-0.1493458	-0.2737411	-0.4253207	-0.5982587	-0.7859080	-0.9810561	-1.1762016	-1.3638427
-1.5367646	-1.6883169	-1.8126673	-1.9050227	-1.9618082					

Ejemplo 4.4.

$$u_{xxxx} + 2u_{xxyy} + u_{yyyy} = 0$$

$$u(x,y) = x^2 + y^2 \; ; \; u_{xx} = u_{yy} = 2, \; (x,y) \in \partial R$$

La solución analítica es $u(x,y) = x^2 + y^2$. El problema es resuelto de acuerdo a (3.40) y (3.41) aplicando el algoritmo para computar Qc. Con n=16 obtuvimos las aproximaciones:

RESULTADOS FINALES

---------- AA ----------

0.0078129	0.0195320	0.0390635	0.0664073	0.1015635	0.1445322	0.1953134	0.2539070	0.3203132	0.3945319
0.4765630	0.5664067	0.6640629	0.7695315	0.8828128					
0.0195320	0.0312512	0.0507829	0.0781267	0.1132830	0.1562517	0.2070328	0.2656263	0.3320323	0.4062509
0.4882820	0.5781256	0.6757817	0.7812504	0.8945315					
0.0390635	0.0507829	0.0703146	0.0976585	0.1328148	0.1757834	0.2265644	0.2851579	0.3515638	0.4257823
0.5078133	0.5976569	0.6953130	0.8007816	0.9140628					
0.0664073	0.0781267	0.0976585	0.1250024	0.1601587	0.2031272	0.2539082	0.3125015	0.3789074	0.4531259
0.5351568	0.6250004	0.7226565	0.8281251	0.9414064					
0.1015635	0.1132830	0.1328148	0.1601587	0.1953149	0.2382834	0.2890642	0.3476575	0.4140633	0.4882818
0.5703127	0.6601563	0.7578124	0.8632812	0.9765625					
0.1445322	0.1562517	0.1757834	0.2031272	0.2382834	0.2812518	0.3320326	0.3906258	0.4570316	0.5312499
0.6132809	0.7031245	0.8007807	0.9062495	1.0195311					
0.1953134	0.2070328	0.2265644	0.2539082	0.2890642	0.3320326	0.3828133	0.4414065	0.5078122	0.5820306
0.6640615	0.7539051	0.8515614	0.9570304	1.0703121					
0.2539070	0.2656263	0.2851579	0.3125015	0.3476575	0.3906258	0.4414065	0.4999996	0.5664053	0.6406236
0.7226546	0.8124983	0.9101547	1.0156238	1.1289056					
0.3203132	0.3320323	0.3515638	0.3789074	0.4140633	0.4570316	0.5078122	0.5664053	0.6328109	0.7070293
0.7890603	0.8789041	0.9765605	1.0820297	1.1953117					
0.3945319	0.4062509	0.4257823	0.4531259	0.4882818	0.5312499	0.5820306	0.6406236	0.7070293	0.7812477
0.8632788	0.9531226	1.0507791	1.1562484	1.2695305					
0.4765630	0.4882820	0.5078133	0.5351568	0.5703127	0.6132809	0.6640615	0.7226546	0.7890603	0.8632788
0.9453098	1.0351537	1.1328102	1.2382795	1.3515616					
0.5664067	0.5781256	0.5976569	0.6250004	0.6601563	0.7031245	0.7539051	0.8124983	0.8789041	0.9531226
1.0351537	1.1249975	1.2226540	1.3281234	1.4414055					
0.6640629	0.6757817	0.6953130	0.7226565	0.7578124	0.8007807	0.8515614	0.9101547	0.9765605	1.0507791
1.1328102	1.2226540	1.3203105	1.4257798	1.5390618					
0.7695315	0.7812504	0.8007816	0.8281251	0.8632812	0.9062495	0.9570304	1.0156238	1.0820297	1.1562484
1.2382795	1.3281234	1.4257798	1.5312489	1.6445308					
0.8828128	0.8945315	0.9140628	0.9414064	0.9765625	1.0195311	1.0703121	1.1289056	1.1953117	1.2695305
1.3515616	1.4414055	1.5390618	1.6445308	1.7578124					

5. PROGRAMAS COMPUTACIONALES.

Los programas computacionales para resolver las ecuaciones de Helmholtz, de Poisson y Biarmónica, con condiciones de Dirichlet en la frontera de $R = [a,b]x[c,d]$, $b-a = d-c$, mediante el método del producto tensorial con uso de la transformada rápida de Fourier, son presentados en el apéndice. Ellos tienen en común las subrutinas IMPRIM y FFTMOR.

La subrutina IMPRIM imprime los resultados finales. Sus parámetros son una matriz B y el número natural N. La subrutina FFTMOR tiene por propósito computar los productos Qc donde Q es la matriz $(\text{sen } \frac{\pi ij}{n})$ de orden $(n-1)x(n-1)$, y c es un vector columna. FFTMOR llama a su vez a la subrutina FFT, correspondiente al algoritmo de Cooley-Tukey para computar la transformada discreta de Fourier, presentado en [1]. Los parámetros de la subrutina FFTMOR son el vector c, $N = 2^k$, NSMALL = 2N, NU = k.

Para el caso de las ecuaciones de Helmholtz y de Poisson el segundo miembro del respectivo sistema de ecuaciones lineales es generado mediante la subrutina SEG.

Los parámetros de esta subrutina son una matriz B, en la cual se almacenan por filas, las componentes de dicho miembro, el natural $N = 2^{k+1}$, los vértices XA, XB, YC y YD del cuadrado R, y el valor H que corresponde a la distancia entre los puntos de la partición. Previamente deben definirse las funciones F(X,Y) y G(X,Y) correspondientes a f(x,y) y a la definición de u(x,y) en la frontera de R, respectivamente.

Para el problema de la ecuación Biarmónica, se segundo miembro es generado mediante la subrutina ASEG, cuyos parámetros son los

mismos de la subrutina SEG. Ahora deben definirse las funiones F(X,Y), G(X,Y), P(X,Y) y Q(X,Y) correspondiente a f(x,y) y los valores de u(x,y), $\frac{\partial u^2}{\partial x^2}$ (x,y) y $\frac{\partial^2 u}{\partial y^2}$ (x,y) en la frontera de R, respectivamente.

APENDICE A: PROGRAMA PARA LA ECUACION DE HELMHOLTZ Y ECUACION DE POISSON (σ=0).

```
      DOUBLE PRECISION V(64,64),T(64,64),R(64,64),S(64,64),G(64,64)
      DOUBLE PRECISION FILA(64),COLUM(64),H,LAMDA,SIGMA,A,B,C,D,HH
      EQUIVALENCE(V(1,1),T(1,1),R(1,1),S(1,1),G(1,1))
      EQUIVALENCE(FILA(1),COLUM(1))
      INTEGER NSMALL,I,J,K,N
      COMMON NSMALL
      OPEN(UNIT=22,FILE='DAT.DAT',STATUS='OLD')
      OPEN(UNIT=8,FILE='SAL.LIS',STATUS='NEW')
C-----------------------------------------------------------------------
CC
CC       Este programa resuelve la ecuacion de HELMHOLTZ y de
CC     POISSON (sigma=0), usando el metodo del producto tensorial
CC     con uso de la transformada rapida de FOURIER.
CC
C-----------------------------------------------------------------------
      READ(22,2)K,A,B,C,D,SIGMA
      NSMALL=2**(K+1)
      N=2**K
      H=(B-A)/FLOAT(NSMALL)
      HH=H**2
      IF(K.GT.5)STOP
      CALL SEG(G,NSMALL,H,A,B,C,D)
      WRITE(8,1)
      CALL IMPRIM(G,NSMALL)
1     FORMAT('1',30X,'SEGUNDO MIEMBRO',/31X,15('-'),//)
2     FORMAT(I8,X,F8.5,X,F8.5,X,F8.5,X,F8.5,X,F8.5)
C
C
      DO 30 I=1,NSMALL-1
         DO 10 J=1,NSMALL-1
10          FILA(J)=G(I,J)
         CALL FFTMOR(FILA,N,NSMALL,K)
         DO 30 J=1,NSMALL-1
30          R(I,J)=FILA(J)
C
C
      DO 50 I=1,NSMALL-1
         DO 40 J=1,NSMALL-1
40          COLUM(J)=R(J,I)
         CALL FFTMOR(COLUM,N,NSMALL,K)
      DO 50 J=1,NSMALL-1
50       S(J,I)=COLUM(J)/(LAMDA(J)+LAMDA(I)+SIGMA*HH)
C
      DO 70 I=1,NSMALL-1
         DO 60 J=1,NSMALL-1
60          FILA(J)=S(I,J)
         CALL FFTMOR(FILA,N,NSMALL,K)
         DO 70 J=1,NSMALL-1
70          T(I,J)=FILA(J)
C
      DO 90 I=1,NSMALL-1
         DO 80 J=1,NSMALL-1
80          COLUM(J)=T(J,I)
         CALL FFTMOR(COLUM,N,NSMALL,K)
         DO 90 J=1,NSMALL-1
90          V(J,I)=COLUM(J)/FLOAT(N**2)
C
  100 WRITE(8,56)
      CALL IMPRIM(V,NSMALL)
   56 FORMAT('0',40X,'RESULTADOS FINALES',/41X,18('-'),//)
      STOP
      END
```

```
C234567
      SUBROUTINE FFTMOR(C,N,NSMALL,NU)
      DOUBLE PRECISION TERM1,TERM2,TERM3,YREAL(32),YIMAG(32)
      DOUBLE PRECISION C(NSMALL),D(64)
      DOUBLE PRECISION Y2REAL(32),Y2IMAG(32),PI,CIMAG(64)
      PI=3.141592654
      YREAL(1)=0
      YIMAG(1)=C(1)

      DO 40 J=2,N
         IPAR=2*J-2
         IMPAR=2*J-1

         YREAL(J)=C(IPAR)
   40    YIMAG(J)=C(IMPAR)
      CALL FFT(YREAL,YIMAG,N,NU)
      DO 70 K=1,N-1
         IPAR=2*K
         TERM1=DSIN((PI*K)/N)*(YIMAG(K+1)+YIMAG(N-K+1))/2.
         TERM2=(YIMAG(K+1)-YIMAG(N-K+1))/2.
         TERM3=DCOS((PI*K)/N)*(YREAL(K+1)-YREAL(N-K+1))/2.
   70    D(IPAR)=TERM3+TERM1-TERM2
C
      DO 80 J=1,NSMALL-1
         CIMAG(J)=C(J)*DSIN((PI*J)/NSMALL)*(-1.0)
   80    C(J)=C(J)*DCOS((PI*J)/NSMALL)
C
      YREAL(1)=0
      YIMAG(1)=0
      Y2REAL(1)=C(1)
      Y2IMAG(1)=CIMAG(1)
C
      DO 90 J=2,N
         IPAR=2*J-2
         IMPAR=2*J-1
         YREAL(J)=C(IPAR)
         YIMAG(J)=CIMAG(IPAR)
         Y2REAL(J)=C(IMPAR)
   90    Y2IMAG(J)=CIMAG(IMPAR)
C
      CALL FFT(YREAL,YIMAG,N,NU)
      CALL FFT(Y2REAL,Y2IMAG,N,NU)
C
      DO 100 K=0,N-1
         TERM1=DSIN((PI*K)/N)*Y2REAL(K+1)
         TERM2=DCOS((PI*K)/N)*Y2IMAG(K+1)
         IMPAR=2*K+1
  100    D(IMPAR)=TERM1-TERM2-YIMAG(K+1)
C
      DO 110 K=1,NSMALL-1
  110    C(K)=D(K)
      RETURN
      END
```

```
C234567
      SUBROUTINE FFT(XREAL,XIMAG,N,NU)
      DOUBLE PRECISION XREAL(N),XIMAG(N),ARG,P,C,S,TREAL,TIMAG
      N2=N/2
      NU1=NU-1
      K=0
      DO 100 L=1,NU
  102    DO 101 I=1,N2
             NAUX=K/2**NU1
             P=IBITR(NAUX,NU)
             ARG=6.283185*P/FLOAT(N)
             C=DCOS(ARG)
             S=DSIN(ARG)
             K1=K+1
             K1N2=K1+N2
             TREAL=XREAL(K1N2)*C+XIMAG(K1N2)*S
             TIMAG=XIMAG(K1N2)*C-XREAL(K1N2)*S
             XREAL(K1N2)=XREAL(K1)-TREAL
             XIMAG(K1N2)=XIMAG(K1)-TIMAG
             XREAL(K1)=XREAL(K1)+TREAL
             XIMAG(K1)=XIMAG(K1)+TIMAG
  101        K=K+1
C
      K=K+N2
      IF(K.LT.N)GO TO 102
      K=0
      NU1=NU1-1
  100 N2=N2/2
      DO 103 K=1,N
         I=IBITR(K-1,NU)+1
         IF(I.LE.K)GO TO 103
         TREAL=XREAL(K)
         TIMAG=XIMAG(K)
         XREAL(K)=XREAL(I)
         XIMAG(K)=XIMAG(I)
         XREAL(I)=TREAL
         XIMAG(I)=TIMAG
  103 CONTINUE
      RETURN
      END
```

```
C234567
      SUBROUTINE SEG(B,N,H,XA,XB,YC,YD)
      DOUBLE PRECISION B(64,64),H,F,G,XA,XB,YC,YD
      DO 10 J=1,N-1
         DO 10 I=1,N-1
   10       B(I,J)=F(XA+I*H,YC+J*H)*(H**2)
      DO 20 K=1,N-1
            B(1,K)=B(1,K)+G(XA+K*H,YC)
   20       B(N-1,K)=B(N-1,K)+G(XA+K*H,YD)
      DO 30 K=1,N-1
            B(K,1)=B(K,1)+G(XA,YC+K*H)
   30       B(K,N-1)=B(K,N-1)+G(XB,YC+K*H)
      RETURN
      END
C234567
      SUBROUTINE IMPRIM(B,N)
      DOUBLE PRECISION B(64,64)
      INTEGER N
      WRITE(8,5)
      DO 10 I=1,N-1
   10    WRITE(8,20)(B(I,J),J=1,N-1)
      RETURN
   20 FORMAT('0',10F13.7)
    5 FORMAT('0',40X,'--------------- ** ---------------')
      END
```

```
C234567
      FUNCTION IBITR(J,NU)
      J1=J
      IBITR=0
      DO 200 I=1,NU
         J2=J1/2
         IBITR=IBITR*2+(J1-2*J2)
  200    J1=J2
      RETURN
      END
C234567
      DOUBLE PRECISION FUNCTION LAMDA(K)
      DOUBLE PRECISION PI
      INTEGER K,NSMALL
      COMMON NSMALL
      PI=3.141592653589
      LAMDA=2.0-(2.0*DCOS((PI*K)/NSMALL))
      RETURN
      END
C234567
      DOUBLE PRECISION FUNCTION F(X,Y)
      DOUBLE PRECISION X,Y,PI
      PI=3.141592653589
      F=2.0*(PI**2)*(DCOS(PI*X)+DCOS(PI*Y))
      RETURN
      END
C234567
      DOUBLE PRECISION FUNCTION G(X,Y)
      DOUBLE PRECISION X,Y,PI
      PI=3.141592653589
      G=DCOS(PI*X)+DCOS(PI*Y)
      RETURN
      END
```

APENDICE B: PROGRAMA PARA LA ECUACION BIARMONICA.

```
      DOUBLE PRECISION V(64,64),T(64,64),R(64,64),S(64,64),G(64,64)
      DOUBLE PRECISION FILA(64),COLUM(64),H,LAMDA,A,B,C,D
      EQUIVALENCE(V(1,1),T(1,1),R(1,1),S(1,1),G(1,1))
      EQUIVALENCE(FILA(1),COLUM(1))
      INTEGER NSMALL,I,J,K,N
      COMMON NSMALL
      OPEN(UNIT=22,FILE='DAT.DAT',STATUS='OLD')
      OPEN(UNIT=8,FILE='SAL.LIS',STATUS='NEW')
C-----------------------------------------------------------------------
CCC                                                                     C
CCC       Este programa resuelve la ecuacion BIARMONICA usando         C
CCC    el metodo del producto tensorial con uso de la transformada     C
CCC    rapida de FOURIER.                                               C
CCC                                                                     C
C-----------------------------------------------------------------------
      READ(22,2)K,A,B,C,D
      NSMALL=2**(K+1)
      N=2**K
      H=(B-A)/FLOAT(NSMALL)
      IF(K.GT.5)STOP
      CALL ASEG(G,NSMALL,H,A,B,C,D)
      WRITE(8,1)
      CALL IMPRIM(G,NSMALL)
1     FORMAT('1',50X,'SEGUNDO MIEMBRO',/51X,15('-'),//)
2     FORMAT(I8,X,F8.5,X,F8.5,X,F8.5,X,F8.5,X,F8.5)
C
C
      DO 30 I=1,NSMALL-1
         DO 10 J=1,NSMALL-1
10          FILA(J)=G(I,J)
         CALL FFTMOR(FILA,N,NSMALL,K)
         DO 30 J=1,NSMALL-1
30          R(I,J)=FILA(J)
C
C
      DO 50 I=1,NSMALL-1
         DO 40 J=1,NSMALL-1
40          COLUM(J)=R(J,I)
         CALL FFTMOR(COLUM,N,NSMALL,K)
      DO 50 J=1,NSMALL-1
50       S(J,I)=COLUM(J)/((LAMDA(J)+LAMDA(I))**2)
C
      DO 70 I=1,NSMALL-1
         DO 60 J=1,NSMALL-1
60          FILA(J)=S(I,J)
         CALL FFTMOR(FILA,N,NSMALL,K)
         DO 70 J=1,NSMALL-1
70          T(I,J)=FILA(J)
C
      DO 90 I=1,NSMALL-1
         DO 80 J=1,NSMALL-1
80          COLUM(J)=T(J,I)
         CALL FFTMOR(COLUM,N,NSMALL,K)
         DO 90 J=1,NSMALL-1
90          V(J,I)=COLUM(J)/FLOAT(N**2)
C
  100 WRITE(8,56)
      CALL IMPRIM(V,NSMALL)
   56 FORMAT('0',//51X,'RESULTADOS FINALES',/51X,18('-'),//)
      STOP
      END
```

```
C234567
      SUBROUTINE FFTMOR(C,N,NSMALL,NU)
      DOUBLE PRECISION TERM1,TERM2,TERM3,YREAL(32),YIMAG(32)
      DOUBLE PRECISION C(NSMALL),D(64)
      DOUBLE PRECISION Y2REAL(32),Y2IMAG(32),PI,CIMAG(64)
      PI=3.141592654
      YREAL(1)=0
      YIMAG(1)=C(1)

      DO 40 J=2,N
         IPAR=2*J-2
         IMPAR=2*J-1

         YREAL(J)=C(IPAR)
   40    YIMAG(J)=C(IMPAR)
      CALL FFT(YREAL,YIMAG,N,NU)
      DO 70 K=1,N-1
         IPAR=2*K
         TERM1=DSIN((PI*K)/N)*(YIMAG(K+1)+YIMAG(N-K+1))/2.
         TERM2=(YIMAG(K+1)-YIMAG(N-K+1))/2.
         TERM3=DCOS((PI*K)/N)*(YREAL(K+1)-YREAL(N-K+1))/2.
   70    D(IPAR)=TERM3+TERM1-TERM2
C
      DO 80 J=1,NSMALL-1
         CIMAG(J)=C(J)*DSIN((PI*J)/NSMALL)*(-1.0)
   80    C(J)=C(J)*DCOS((PI*J)/NSMALL)
C
      YREAL(1)=0
      YIMAG(1)=0
      Y2REAL(1)=C(1)
      Y2IMAG(1)=CIMAG(1)
C
      DO 90 J=2,N
         IPAR=2*J-2
         IMPAR=2*J-1
         YREAL(J)=C(IPAR)
         YIMAG(J)=CIMAG(IPAR)
         Y2REAL(J)=C(IMPAR)
   90    Y2IMAG(J)=CIMAG(IMPAR)
C
      CALL FFT(YREAL,YIMAG,N,NU)
      CALL FFT(Y2REAL,Y2IMAG,N,NU)
C
      DO 100 K=0,N-1
         TERM1=DSIN((PI*K)/N)*Y2REAL(K+1)
         TERM2=DCOS((PI*K)/N)*Y2IMAG(K+1)
         IMPAR=2*K+1
  100    D(IMPAR)=TERM1-TERM2-YIMAG(K+1)
C
      DO 110 K=1,NSMALL-1
  110    C(K)=D(K)
      RETURN
      END
```

```
C234567
      SUBROUTINE FFT(XREAL,XIMAG,N,NU)
      DOUBLE PRECISION XREAL(N),XIMAG(N),ARG,P,C,S,TREAL,TIMAG
      N2=N/2
      NU1=NU-1
      K=0
      DO 100 L=1,NU
  102    DO 101 I=1,N2
             NAUX=K/2**NU1
             P=IBITR(NAUX,NU)
             ARG=6.283185*P/FLOAT(N)
             C=DCOS(ARG)
             S=DSIN(ARG)
             K1=K+1
             K1N2=K1+N2
             TREAL=XREAL(K1N2)*C+XIMAG(K1N2)*S
             TIMAG=XIMAG(K1N2)*C-XREAL(K1N2)*S
             XREAL(K1N2)=XREAL(K1)-TREAL
             XIMAG(K1N2)=XIMAG(K1)-TIMAG
             XREAL(K1)=XREAL(K1)+TREAL
             XIMAG(K1)=XIMAG(K1)+TIMAG
  101        K=K+1
C
      K=K+N2
      IF(K.LT.N)GO TO 102
      K=0
      NU1=NU1-1
  100 N2=N2/2
      DO 103 K=1,N
         I=IBITR(K-1,NU)+1
         IF(I.LE.K)GO TO 103
         TREAL=XREAL(K)
         TIMAG=XIMAG(K)
         XREAL(K)=XREAL(I)
         XIMAG(K)=XIMAG(I)
         XREAL(I)=TREAL
         XIMAG(I)=TIMAG
  103 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE ASEG(B,N,H,XA,XB,YC,YD)
      DOUBLE PRECISION B(64,64),H,F,G,XA,XB,YC,YD
      HH=H**2
      HHH=HH**2
      DO 10 J=1,N-1
         DO 10 I=1,N-1
   10       B(I,J)=F(XA+I*H,YC+J*H)*HHH
      DO 20 K=1,N-1
            B(1,K)=B(1,K)+G(XA+K*H,YC)*6-G(XA+(K+1)*H,YC)*2
            B(1,K)=B(1,K)-Q(XA+K*H,YC)*HH-G(XA+(K-1)*H,YC)*2
            B(N-1,K)=B(N-1,K)+G(XA+K*H,YD)*6-G(XA+(K+1)*H,YD)*2
   20       B(N-1,K)=B(N-1,K)-Q(XA+K*H,YD)*HH-G(XA+(K-1)*H,YD)*2
      DO 30 K=1,N-1
            B(K,1)=B(K,1)+G(XA,YC+K*H)*6-G(XA,YC+(K+1)*H)*2
            B(K,1)=B(K,1)-G(XA,YC+(K-1)*H)*2-P(XA,YC+K*H)*HH
            B(K,N-1)=B(K,N-1)+G(XB,YC+K*H)*6-G(XB,YC+(K+1)*H)*2
   30       B(K,N-1)=B(K,N-1)-P(XB,YC+K*H)*HH-G(XB,YC+(K-1)*H)*2
C
      B(1,1)=B(1,1)+2*G(XA,YC)
      B(1,N-1)=B(1,N-1)+2*G(XB,YC)
      B(N-1,1)=B(N-1,1)+2*G(XA,YD)
      B(N-1,N-1)=B(N-1,N-1)+2*G(XB,YD)
      B(1,2)=B(1,2)-G(XA,YC+H)
      B(2,1)=B(2,1)-G(XA+H,YC)
      B(1,N-2)=B(1,N-2)-G(XB,YC+H)
      B(N-2,1)=B(N-2,1)-G(XA+H,YD)
      B(N-1,2)=B(N-1,2)-G(XA,YD-H)
      B(2,N-1)=B(2,N-1)-G(XB-H,YC)
      B(N-1,N-2)=B(N-1,N-2)-G(XB,YD-H)
      B(N-2,N-1)=B(N-2,N-1)-G(XB-H,YD)
C
      DO 40 K=2,N-2
            B(2,K)=B(2,K)-G(XA+K*H,YC)
   40       B(N-2,K)=B(N-2,K)-G(XA+K*H,YD)
C
      DO 50 K=2,N-2
            B(K,2)=B(K,2)-G(XA,YC+K*H)
   50       B(K,N-2)=B(K,N-2)-G(XB,YC+K*H)
C
      RETURN
      END
```

```
C234567
      SUBROUTINE IMPRIM(B,N)
      DOUBLE PRECISION B(64,64)
      INTEGER N
      WRITE(8,5)
      DO 10 I=1,N-1
   10    WRITE(8,20)(B(I,J),J=1,N-1)
      RETURN
   20 FORMAT('0',10F13.7)
    5 FORMAT('0',40X,'--------------- ** ---------------')
      END
C234567
      FUNCTION IBITR(J,NU)
      J1=J
      IBITR=0
      DO 200 I=1,NU
         J2=J1/2
         IBITR=IBITR*2+(J1-2*J2)
  200    J1=J2
      RETURN
      END
C234567
      DOUBLE PRECISION FUNCTION LAMDA(K)
      DOUBLE PRECISION PI
      INTEGER K,NSMALL
      COMMON NSMALL
      PI=3.141592653589
      LAMDA=2.0-(2.0*DCOS((PI*K)/NSMALL))
      RETURN
      END
C234567
      DOUBLE PRECISION FUNCTION F(X,Y)
      DOUBLE PRECISION X,Y
      F=0
      RETURN
      END
C234567
      DOUBLE PRECISION FUNCTION G(X,Y)
      DOUBLE PRECISION X,Y
      G=X**2+Y**2
      RETURN
      END
C234567
      DOUBLE PRECISION FUNCTION P(X,Y)
      DOUBLE PRECISION X,Y
      P=2.0
      RETURN
      END
C234567
      DOUBLE PRECISION FUNCTION Q(X,Y)
      DOUBLE PRECISION X,Y
      Q=2.0
      RETURN
      END
```

BIBLIOGRAFIA.

[1] BRIGHAM, E. ORAN: The Fast Fourier Transform
Prentice-Hall, Inc. U.S.A. 1974.

[2] HOUSTIS, E.N.;
PAPATHEODOROU. : Comparison of Fast Direct Methods for Elliptic Problems.
Advances in Computer Methods for Partial Differential Equations-II. IMACS(AICA), 1977.

[3] LYNCH, E.; RICE, JOHN R.;
THOMAS, DONALD H.: Direct Solution of Partial Difference Equations by Tensor Product Methods Numer. Math. 6 pp. 185-199, 1964.

[4] LYNCH, E.; RICE, JOHN R.;
THOMAS, DONALD H.: Tensor Product Analysis of Partial Difference Equations, Bull. Amer. Math. Soc. 70, pp. 378-384, 1964.

[5] ROJO, O; SOTO, R.;
VALDIVIA, L. : Aplicación del Método de Descomposición Matricial y la Transformada Rápida de Fourier a la solución de la Ecuación de Poisson.
Revista Proyecciones N° 4, Depto. Matemáticas, U. del Norte, 1983.