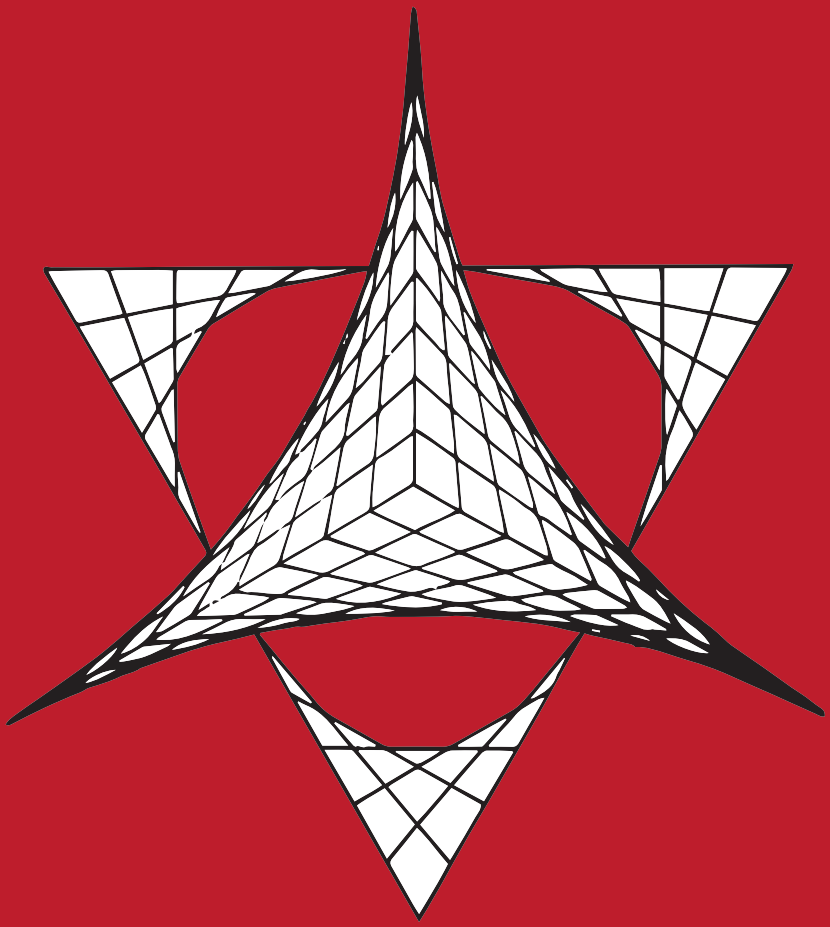
CONFERENCIAS DE ANALISIS NUMERICO

COMPILADOR:
DIEGO BRICIO HERNANDEZ

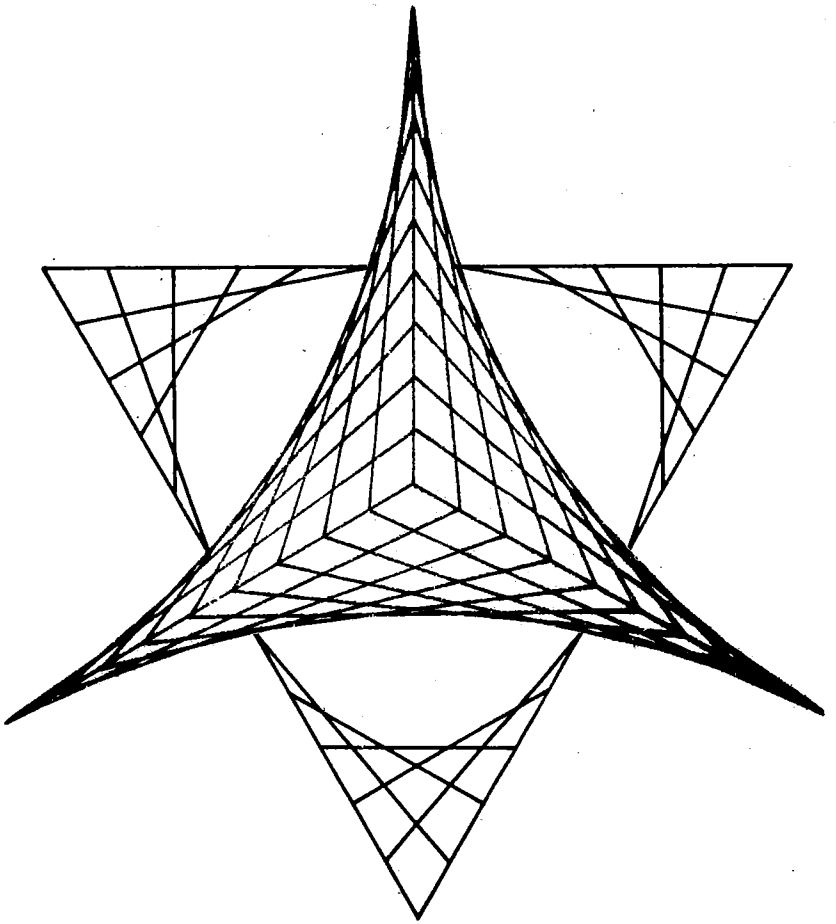


3er COLOQUIO DEPARTAMENTO DE MATEMATICAS
CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL I.PN
LA TRINIDAD¨. TLAXCALA, MÉXIC, AGOSTO DE 1983

CONFERENCIAS DE ANALISIS NUMERICO

COMPILADOR:

DIEGO BRICIO HERNANDEZ



3er. COLOQUIO DEL DEPARTAMENTO DE MATEMATICAS
CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS DEL IPN
LA TRINIDAD, TLAXCALA, AGOSTO DE 1983

3er COLOQUIO
DEPARTAMENTO DE MATEMATICAS
CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS DEL IPN

"La Trinidad", Tlaxcala, México
8 al 26 de Agosto de 1983

PATROCINADORES

Subsecretaría de Educación Superior e Investigacion Científica, S.E.P.
Subsecretaría de Educación e Investigación Tecnológica, S.E.P.
Dirección Adjunta de Desarrollo Científico, CONACyT
Instituto Politécnico Nacional
Centro de Investigación y de Estudios Avanzados del I.P.N.

LAS PRESENTES CONFERENCIAS FUERON DICTADAS EN EL CURSO DE ANÁLISIS NUMÉRICO, QUE SE DESARROLLÓ - DURANTE EL 3ER. COLOQUIO DEL DEPARTAMENTO DE MATEMÁTICAS DEL CINVESTAV.

LA IMPRESIÓN DE ESTAS NOTAS FUE PARCIALMENTE SUBVENSIONADA POR EL DEPARTAMENTO DE MATEMÁTICAS DE LA UNIVERSIDAD AUTÓNOMA METROPOLITANA-IZTAPALAPA BAJO EL PROYECTO PRONAES NO. 2113-003491.

DR. HORACIO TAPIA RECILLAS

COORDINADOR DEL 3ER COLOQUIO

I N D I C E

INTRODUCCION

En agosto de 1983 tuvo lugar el 3er. Coloquio del Departamento de Matemáticas del Centro de Investigación y de Estudios Avanzados del IPN. Ahí se incluyó un curso de Análisis Numérico impartido por el autor de estas líneas, curso cuyas notas aparecieron en su oportunidad dentro de esta misma serie de publicaciones.

Dicho material fue enriquecido con la aportación de ocho conferencias invitadas, en las cuales se exploraban diversos aspectos del tema y se cubrían tanto aspectos teóricos como aplicativos. Las conferencias fueron impartidas por Virginia Abrin, Hans Fetter, Erick Gamas, Hugo Martínez, Luis Mier y Terán, Patricia Saavedra, Carlos Velarde y Luis Verde Star. El presente volumen recoge las versiones escritas de esas pláticas, junto con otro artículo introductorio en el que pretendemos, entre otras cosas, situar dentro del campo del Análisis Numérico a las conferencias en cuestión.

Convencidos del valor del material expuesto por los conferencistas invitados, hemos decidido ponerlo a la disposición de la comunidad matemática mexicana.

Diego Bricio Hernández

Cuernavaca, Enero de 1985

CALCULOS, ALGORITMOS Y APROXIMACION*

Diego Bricio Hernández**

Resumen

Se describen algunos procedimientos para diseñar algoritmos numéricos, además de hacer evidente la necesidad de recurrir a dichos algoritmos siempre que se desee calcular. Por otro lado, se identifica el problema de calcular con el de aproximar, mismo que se discute con cierto detenimiento. Se busca también servir como introducción a los ocho artículos que componen el resto de este volumen, situándolos en el campo del Análisis Numérico.

1. La necesidad de calcular.

Un problema algebraico que se presenta muy frecuentemente es el de resolver sistemas de ecuaciones lineales, como

$$Ax = b, \tag{1}$$

donde $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$. Al respecto, se sabe que (1) tiene solución si y solo si el rango de A no se altera al aumentarla

---

* Este artículo recoge y actualiza el material de dos pláticas impartidas por el autor como parte del curso de Análisis Numérico, del 3er. Coloquio del Departamento de Matemáticas del CIEA-IPN, La Trinidad, Tlax., Agosto 1983.

** Departamento de Matemáticas, Universidad Autónoma Metropolitana, Iztapalapa, Apdo. Postal 55-534, 09340 México, D.F.

a n + 1 columnas agregando b [3]. Más aún, la solución es única si y solo si m = n y, además, el determinante de A no es cero. En ese caso se dispone de un algoritmo, la regla de Cramer, para encontrar la solución.

En Análisis, ocurre a menudo el problema de calcular la integral de una función f, a saber

$$\int_a^b f(x)dx \tag{2}$$

Se sabe que la integral (2) existe y es única si, por ejemplo, f es continua sobre [a,b]. En ese caso, la regla de Barrow nos da un algoritmo muy conveniente para calcular (2), en la forma de

$$\int_a^b f(x)dx = F(b) - F(a),$$

donde F es tal que f' = f.

En Mecánica del Medio Continuo se estudian sistemas como la cuerda vibrante. Bajo suposiciones como:

- la cuerda es homogenea, ligera y muy delgada
- las oscilaciones son pequeñas
- la tensión a lo largo de la cuerda es uniforme
- la fricción con el aire es despreciable
- cada punto de la cuerda vibra a lo largo de la vertical
- los extremos de la cuerda están fijos

se llega facilmente [4] a que la amplitud de las oscilaciones

satisface la ecuación en derivadas parciales

$$(3') \qquad \frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \quad (t < 0,\ 0 < x < \ell),$$

además de que

$$(3'') \qquad u(t,0) = 0 \qquad u(t,\ell) = 0$$

Bajo estas circunstancias, se demuestra por el método de separación de variables [4] que la solución de (3) correspondiente a la condición inicial

$$u(0,x) = \bar{u}(x) \qquad u_t(0,x) = 0$$

es precisamente

$$(4) \qquad u(t,x) = \sum_{k=1}^{\infty} A_k \cos \frac{k\pi ct}{\ell} \operatorname{sen} \frac{k\pi x}{\ell}$$

En (4),

$$A_1, A_2, \ldots$$

son los coeficientes de Fourier de $\bar{u}$ con respecto al sistema ortogonal

$$\operatorname{sen} \frac{\pi x}{\ell}\ ,\ \operatorname{sen} \frac{2\pi x}{\ell}\ , \ldots$$

En particular, si nos interesa la trayectoria del punto

medio de la cuerda, debemos sumar la serie

$$(5) \qquad \sum_{k=0}^{\infty} \alpha_k \cos(2k+1)\,\frac{\pi c t}{\ell}$$

donde $\alpha_k = (-1)^k A_{2k+1}$, $k = 0,1,2,\ldots$

Sin embargo, basta relajar algunas de las hipótesis que subyacen a (3) para que la situación cambie grandemente. Por ejemplo si eliminamos la suposición de que las vibraciones son pequeñas, en lugar de (3') se obtiene

$$(6) \qquad \frac{\partial^2 x}{\partial t^2} = c^2 \frac{\partial}{\partial x}\left(\frac{u_x}{\sqrt{1+u_x^2}} \right)$$

Ahora bien, esta última ecuación en derivadas parciales es no lineal y no es posible resolverla en forma explícita, aún cuando haya teoremas que garantizan la existencia de una sola solución dadas las condiciones (3") y (4). En este caso no queda sino aplicar un método numérico para aproximar dicha solución; un método de este tipo reduce el problema de resolver la ecuación (6) sujeta a (3") y (4) al problema más simple de resolver un sistema de ecuaciones algebraicas no lineales.

Resulta, pues, que para resolver formulaciones "realistas", a menudo no queda sino recurrir a los métodos numéricos.

Al hacerlo, debemos por fuerza perder precisión en los cálculos, pues todo método numérico se basa en aproximaciones de uno u otro tipo. Sin embargo, siempre y cuando se mantenga

el error de aproximación bajo control no habrá ningún problema.

Ya hemos visto que los métodos clásicos nos ofrecen algoritmos para tratar al menos las formulaciones simplificadas como (3'). Sin embargo, una segunda ojeada a fórmulas como (4) nos revela que de explícitas no tienen mucho cuando se trata de calcular. Por ejemplo, si se trata de evaluar (5) para c, ℓ y t fijos debemos primero calcular la sucesión

$$u_k := a_k \cos(2k+1)\pi ct/\ell \tag{7}$$

y luego sumar la serie

$$S := \sum_{k=0}^{\infty} u_k \tag{8}$$

Suponiendo que los términos de $\{u_k\}$ se puedan calcular exactamente, queda todavía el problema de sumar la serie (8). A menos que por algún truco ingenioso se logre expresar S en forma trambién analítica, en general el cálculo de S involucrará necesariamente algún tipo de aproximación: basta pensar en que no podemos ni siquiera calcular una infinidad de términos $(u_1, u_2, \ldots)$, mucho menos sumarlos. Bajo estas circunstancias, el cálculo de S se efectuará mediante un algoritmo similar al siguiente:

i) Dado $\varepsilon > 0$;

ii) $S := u_1$; $n := 2$;

iii) Mientras $(|u_n|/S > \varepsilon)$

$S := S + u_n;$

$n := n + 1;$

iv) Fin.

Esto, desde luego, involucra alguna pérdida de precisión, pero llevamos un control de ello mediante la tolerancia ε y entonces no hay problema.

El artículo de Hans Fetter ofrece una metodología para escribir "programas" como el anterior, de tal manera que se puedan describir algoritmos que resuelvan problemas como éste y más complejos.

Y, ¿qué sucede con problemas más simples, como puede ser el cálculo de (2)? La regla de Barrow dice que

$$\int_0^1 xe^{-x^2/2}\,dx = 1 - \frac{1}{\sqrt{e}}\,. \tag{9}$$

Pero, ¿y si se desea calcular $\int_0^1 e^{-x^2/2}\,dx$? El teorema fundamental del cálculo dice que una antiderivada de $e^{-x^2/2}$ está dada por

$$F(x) = \int_0^x e^{-\xi^2/2}\,d\xi,$$

pero eso no ayuda mucho. Aun cuando sí exista una antiderivada expresable en términos de funciones elementales, eso no implica necesariamente que los cálculos con ellas sean explí-

citos. Piénsese, por ejemplo, en los problemas que implica calcular el segundo miembro de (9). Sin duda, también aquí hay que aprender a vivir con aproximaciones y por lo tanto pérdida de precisión, por lo que pudiera ser conveniente utilizar un método numérico desde el inicio.

Un método numérico para el cálculo de (2) busca evaluar una suma de la forma

$$\sum_{i=1}^{N} \rho_i f(x_i) \tag{10}$$

donde $N, x_1, \ldots, x_N \in [a,b]$ y $\rho_1, \ldots, \rho_N$ son dados e influyen en la precisión del cálculo.

Y ¿qué se puede decir acerca del problema (1)? Desde luego que la regla de Cramer es adecuada para $n \leq 3$ pero no más allá: evaluar un determinante de orden n requiere calcular $n!(n-1)$ multiplicaciones, decidir si se cambia el signo o no un total de $n!$ veces y calcular $n!-1$ sumas. Por si fuera poco, la regla de Cramer requiere que se calculen $n+1$ determinantes de orden n y que se efectúen n divisiones. Un cálculo sencillo muestra que, trabajando a razón de 10^6 operaciones/seg., una computadora tardaría 132,560 años en resolver un sistema de 20 ecuaciones lineales usando la regla de Cramer. Así pues, no es un método práctico para sistemas grandes.

Por otro lado, el método de eliminación gaussiana, tambien clásico, sí resulta práctico aún para n moderadamente grande (digamos $n \gtrsim 100$), pues requiere del orden de n^3 opera-

ciones. Sin embargo hay aplicaciones –como por ejemplo la solución numérica de problemas de valores iniciales y a la frontera asociados a (6)– para los que deben resolverse sistemas como (1) pero con n del orden de miles. Aquí entran ya consideraciones de eficiencia en los cálculos, que son determinantes no sólo en la organización de los mismos sino aún en la mera posibilidad de almacenar las matrices involucradas. El artículo de Virginia Abrín se ocupa de algunas de estas cuestiones.

2. Diseño de algoritmos numéricos.

Volvamos por un momento al cálculo de los términos de la sucesión $\{u_k\}$, requisito indispensable para la evaluación de S en (8). En cada etapa del cálculo hay que obtener valores de la función cos. Por supuesto que para ello disponemos de la definición geométrica del coseno, pero quizá sea más conveniente pensar en métodos de cálculo que recurran a las operaciones elementales de la aritmética y nada más. Por ejemplo, las series de potencias, pues sabemos que

$$\cos t = \sum_{k=0}^{\infty} (-1)^k \frac{t^{2k}}{(2k)!} \tag{11}$$

De nuevo hay que sumar una serie, por lo que basta redefinir la sucesión $\{u_k\}$ adecuadamente y calcular la serie (8) mediante el algoritmo propuesto para ello. Sin embargo, puede

resultar ineficiente sumar una serie al calcular cada uno de los términos de otra serie que es la que realmente se desea sumar. Una alternativa es aproximar cos mediante una función g que sea "más simple", en el sentido de que pueda calcularse sin recurrir a nada más que las operaciones aritméticas. Por ejemplo, sabemos que la serie en (11) converge uniformemente en compactos y que, gracias a las propiedades de cos, podemos restringir t a $[0,\pi/2]$. En ese caso, dado $\delta > 0$ existe $N \geq 1$ tal que

$$\sup_{0 \leq t \leq \pi/2} |\cos t - p_N(t)| < \delta, \tag{12}$$

donde

$$p_N(t) := \sum_{k=0}^{N} (-1)^k \frac{t^{2k}}{(2k)!}$$

Esta situación ya es más satisfactoria, pues N es fijo, pero todavía hay problemas: N puede ser prohibitivamente grande. La teoría de Aproximación [5] busca encontrar métodos para dar aproximaciones (por ejemplo polinomiales, aunque no necesariamente) de manera de lograr condiciones del tipo de (12) pero con un número manejable de términos. Véase el artículo de Luis Verde Star al respecto.

En particular, dada una función f en (2) que resulta "difícil" de integrar, puede reemplazarse por otra función g que sea "más simple" que ella —en el sentido de que su inte-

gral se evalúa más fácilmente con los medios a nuestra disposición. Es de esta manera que se obtienen las fórmulas de integración numérica como (10): de hecho,

$$\int_a^b g(x)dx = \sum_{i=1}^{N} \rho_i f(x_i).$$

Pero, volvamos por un momento al cálculo de cos t para $0 < t < \pi/2$. Sabemos que cos resuelve el problema de valores iniciales

$$\ddot{\theta} + \theta = 0 \qquad \theta(0) = 1 \qquad \dot{\theta}(0) = 1 \tag{13}$$

y que hay buenos métodos para resolver este tipo de problemas. ¿No será más conveniente calcular la solución de (13) en el instante t directamente, por algún método numérico y utilizarla como aproximación de cos t ? Podríamos proceder de la siguiente manera:

En la forma usual, (13) es equivalente al sistema lineal

$$\begin{pmatrix} \dot{\theta} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} \theta \\ w \end{pmatrix} \qquad \begin{pmatrix} \theta(0) \\ w(0) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

que es de la forma

$$\dot{x} = Ax \qquad x(0) = \bar{x}, \tag{14}$$

cuya solución está dada por

$$x(t) = e^{At}\bar{x}. \tag{15}$$

Si observamos que $(1 - 1/2\,z)^{-1}(1 + 1/2\,z)$ es una aproximación de Padé para e^z, válida para $z > 0$ [2], podemos aproximar el segundo miembro de (15) por $(I - \frac{1}{2}\,hA)^{-1}(I + \frac{1}{2}\,hA)\bar{x}$. En particular, si dividimos el intervalo de cálculo [a,b] según la partición

$$0 = t_0 < t_1 < \ldots < t_N = \pi/2, \tag{16'}$$

a la que corresponden los valores

$$\bar{x} =: x^{(0)}, x^{(1)}, \ldots, x^{(N)} \tag{16"}$$

de la solución de (14), entonces obtenemos que

$$x^{(k+1)} := (I - \tfrac{1}{2}\,hA)^{-1}(I + \tfrac{1}{2}\,hA)x^{(k)}$$

nos permite encontrar una aproximación a (16") de manera iterativa.

En efecto, para nuestro problema la relación anterior se especializa a

$$\theta_{n+1} := \theta_k + \gamma w_k \qquad \theta_0 = 1 \tag{17'}$$

$$w_{k+1} := -\gamma\theta_k + \nu w_k \qquad w_0 = 0, \tag{17"}$$

donde

$$\gamma := \frac{h}{1 + h^2/4} \qquad \nu := \frac{1 - h^2/4}{1 + h^2/4}$$

y por simplicidad hemos tomado una partición uniforme en (16').

Calculados $\theta_1,\ldots,\theta_N$, los almacenamos y entonces calculamos cos t interpolando los puntos $(t_0,\theta_0),(t_1,\theta_1),\ldots,(t_N,\theta_N)$. Alternativamente, si el problema requiere aproximar cos t a m decimales se puede proceder eligiendo N suficientemente grande como para que

$$|\cos t - \cos t_k| < 10^{-m} \quad \text{si} \quad t_k \le t \le t_{k+1}$$

y entonces no hay que interpolar: hasta tomar θ_k como el valor de cos t. Una estrategia parecida a ésta permite a Carlos Velarde en su artículo dar un algoritmo similar a (17) y que resulta sumamente versátil.

El problema de evaluar cos t nos ha llevado a proponer algoritmos de cálculo siguiendo los dos caminos siguientes:

i) aproximar cos mediante una función "más simple" y usar ésta para el cálculo;

ii) inventar un problema equivalente al de calcular cos t y resolverlo independientemente.

A manera de conclusión, podemos señalar dos caminos que se utilizan muy frecuentemente para diseñar algoritmos numéricos:

I.- Aproximar la función que se desea integrar o evaluar, la ecuación que se desea resolver, etc. mediante una función o una ecuación que sea "más simple", en el sentido de que esté más a nuestro alcance procesar. Luego es este objeto más simple el que se procesa.

II.- Inventar un nuevo problema, equivalente al primero, pero "más fácil" de resolver. Luego es este problema el que se resuelve

Ni que decir que en la práctica funciona mejor combinar estos dos principios, en lugar de mantenerlos por separado. Por ejemplo, al decidir calcular cos t como un valor de la solución de (13) utilizamos II, pero al reducir (13) a la iteración (17) hemos hecho uso de I.

Ya hemos indicado cómo es que I da lugar a un algoritmo para calcular la integral (2). Cerraremos esta sección ejemplificando el uso de II para el mismo fin. Comencemos por suponer que a = 0, b = 1, de tal manera que se trata del cálculo de

$$I := \int_0^1 f(x)dx,$$

número al que unicamente le pedimos que esté definido, lo cual permite una clase bastante grande de integrandos.

Sea U una variable aleatoria uniformemente distribuida en $[0,1]$ y sea $Y := f \circ U$, tambien variable aleatoria. Entonces, I es el valor esperado de Y, al que denotamos por EY. Así pues, el cálculo de I se reduce al del valor esperado de una variable aleatoria. Para ello contamos con la ley fuerte de los números grandes, según la cual

$$\lim_{N \to \infty} \frac{Y_1 + \ldots + Y_N}{N} = EY$$

con probabilidad 1, siempre que $Y_1,\ldots,Y_N$ sean muestras independientes de Y.

Una manera de implantar lo anterior consiste en generar números aleatorios $u_1,u_2,\ldots$ independientes y uniformemente distribuidos en [0,1], para aplicar el siguiente algoritmo, muy parecido al descrito en la sección anterior:

i) $y_1 := f(u_1)$; $n := 1$; $I := 0$;

ii) Mientras $(|y_n| \geq \varepsilon n)$

$I := (1 - 1/n)I + y_n/n$;

$n := n + 1$

$y_n := f(u_n)$

iii) Fin

Este procedimiento para al arrojar un valor de la integral que resulta correcto y con un error no mayor que ε, con probabilidad 1.

Este tipo de algoritmos constituyen el método de Monte Carlo. En la monografía [6] se presenta el desarrollo de este enfoque, aplicado a la resolución de problemas tomados de un gran número de campos.

3. Aproximación.

Supongamos que se desea calcular la integral de (9) usando aritmética de m cifras decimales de precisión; en otras palabras, se desea encontrar un número de la forma $0.b_1\ldots b_m$,

con $b_i \in \{0,1,\ldots,g\}$, $i = 1,\ldots,m$ tal que

$$(18) \qquad \left| \frac{1}{\sqrt{e}} - 0.b_1\ldots b_m \right| < \frac{1}{10^{m+1}}$$

Suponiendo que e puede calcularse de manera exacta hasta un número suficiente p de cifras, es bien sabido que (18) puede lograrse implantando el siguiente algoritmo (Newton):

i) y dado;

ii) $x := \frac{1}{2}\left(y + \frac{1}{ey}\right)$;

iii) Mientras $(|y - x| \geq 10^{-m-1})$

$y := x$

$x := \frac{1}{2}\left(y + \frac{1}{ey}\right)$

iv) Fin

Este algoritmo proporciona una receta para aproximar el objeto "complejo" $1/\sqrt{e}$ mediante un objeto "más simple", de la forma $0.b_1\ldots b_m$.

En general, el cálculo de un número $r \in (0,1)$ con m cifras decimales de precisión implica diseñar un algoritmo que, en un número finito de pasos, arroje un número $0.c_1\ldots c_m$, con $c_i \in \{0,1,\ldots,9\}$, $i = 1,\ldots,m$, tal que

$$|r - 0.c_1\ldots c_m| < \frac{1}{10^{m+1}}$$

Ahora bien, los números de la forma $0.c_1\ldots c_m$ constituyen un conjunto (finito) de números reales, por lo que se cae en un caso particular del siguiente

Problema. Sea I el intervalo $(0,1)$ y sea $F \subset I$. Para cada $\varepsilon > 0$, dar una función $c: I \to F$ tal que

$$|\alpha - c(\alpha)| < \varepsilon \qquad \forall\, \alpha \in I$$

En los términos usuales, este problema se refiere a calcular los números en (0,1) mediante los números "más simples" en F; c no es sino el algoritmo que resuelve el problema. En la práctica, basta saber calcular los números en (0,1) para que podamos calcular cualquier real.

Por otro lado, el problema anterior es un caso particular del siguiente, cuyo interés es claro en vista del procedimiento I visto en la sección anterior:

Problema de aproximación. Sea Φ un espacio métrico, con distancia d y sea $\Sigma \subset \Phi$, $\Sigma \neq \phi$. Para cada $\varepsilon > 0$, dar una función $ap: \Phi \to \Sigma$ tal que

$$d(f, ap(f)) < \varepsilon \qquad \forall\, f \in \Phi$$

Como para cualquier otro problema, debemos hacernos dos preguntas:

i) ¿Tiene solución?

ii) ¿Cómo encontrar al menos una?

En el mejor de los casos, habría también que caracterizar el conjunto de soluciones; por ejemplo, ¿cuándo hay una sola?

Por ejemplo, sea Φ el espacio $\mathbb{R}^n$ con la métrica euclidiana usual y sea Σ un subespacio. El teorema de proyección

ortogonal [9] dice que hay una sola función $P:\Phi \to \Sigma$, lineal y tal que $P(\Phi) = \Sigma$, además de que

$$|x - P(x)| = \inf_{z \in \Sigma} |x - z|;$$

dicha función está caracterizada por la propiedad de que

$$x - P(x) \perp \Sigma \qquad \forall\, x \in \Phi$$

Vemos entonces que existe $\varepsilon_0 > 0$ tal que si $\varepsilon < \varepsilon_0$ entonces no hay solución del problema de aproximación correspondiente, en tanto que para $\varepsilon \geq \varepsilon_0$ hay una sola, P. En este caso es posible calcular (es decir, aproximar) los vectores de $\mathbb{R}^n$ mediante los del subespacio Σ.

En otros casos la situación es menos simple. Por ejemplo, sea Φ la clase de todas las funciones continuas en [a,b] y sea Σ la de los polinomios, con

$$d(f,g) = \sup_{a \leq t \leq b} |f(t) - g(t)| \tag{19}$$

El Teorema de Aproximación de Weiestrass [8] dice que el problema de aproximación correspondiente sí tiene solución, pero ni dice cuantas hay ni da un método para encontrar una. Sin embargo, un tal método existe y para simplificar su descripción supondremos que a = 0, b = 1, sin pérdida de generalidad.

Fijemos $n \geq 1$, $0 \leq t \leq 1$ y sea X una variable aleatoria binomial de parámetros n y t, es decir

$$P(X = k) = \binom{n}{k} t^k (1-t)^{n-k} \qquad k = 0,1,\ldots,n$$

Entonces $X/n \in [0,1]$ con probabilidad 1, por lo que $f(X/n)$ es una variable aleatoria bien definida y acotada, cualquiera que sea $f \in \Phi$. Sea $B_n(f,t)$ la esperanza de dicha variable aleatoria, es decir,

$$B_n(f,t) = \sum_{k=0}^{n} f(k/n) \binom{n}{k} t^k (1-t)^{n-k}$$

Claramente $B_n(f,\cdot)$ es un polinomio; se le llama el enésimo polinomio de Bernstein correspondiente a f. Puede demostrarse [1] que

$$d(f,B_n(f,\cdot)) < \omega(f,1/\sqrt{n})$$

donde $\omega: \Phi x (0,\infty) \to [0,\infty)$ es el módulo de continuidad, definido por

$$\omega(f,\delta) := \sup_{A_\delta} |f(s)-f(t)|$$

con

$$A_\delta := \{(s,t) \in [0,1]^2 : |s-t| \leq \delta\}$$

La continuidad uniforme de f asegura que, dado $\varepsilon > 0$, existe $N \geq 1$ tal que

$$|s-t| < \frac{1}{\sqrt{N}} \Longrightarrow |f(s)-f(t)| < \varepsilon$$

Evidentemente, basta definir

$$ap(f) := B_N(f;\cdot)$$

para tener una solución del problema de aproximación correspondiente. Falta todavía ver que N no sea prohibitivamente alto para que esta solución sea de hecho práctica.

Por otro lado, la interpolación es una herramienta muy poderosa para resolver el problema de aproximación cuando Φ consta de funciones definidas en un conjunto D. La idea es que si dos funciones f,g coinciden en un subconjunto "suficientemente grande" S de D, es "poco probable" que d(f,g) sea grande. Para precisar, sea D = [a,b] y sea S una partición finita $a =: x_0 < x_1 < \ldots < x_N < x_{N+1} := b$ de [a,b]

Problema de interpolación. Encontrar $int:\Phi \to \Sigma$ tal que

$$int(f)|_S = f|_S \qquad \forall f \in \Phi$$

Es bien conocida la teoría asociada al problema de interpolación cuando las funciones son reales y Σ consta de todos los polinomios; hay solución única y, además, int(f) es un polinomio de grado no mayor que N+1, cualquiera que sea f. Véase el artículo de Luis Verde Star al respecto de cómo construir int en varios casos de interés.

Por lo pronto, diremos que no es muy conveniente trabajar con polinomios de grado muy alto, pues oscilan mucho y entonces puede suceder que int(f) no sea aceptable como solución

del problema de aproximación. Ahora bien, si Φ consta de todas las funciones continuas en $[a,b]$, con d como en (19) y si sólo se desea calcular integrales como $\int_a^b f(x)dx$, para $f \in \Phi$, entonces basta con que Σ sea cualquier subconjunto de Φ, no necesariamente las funciones polinomiales. Siguiendo el procedimiento I de la sección anterior, se calcula $\int_a^b ap(f)$ en lugar de $\int_a^b f$. Entonces

$$\left|\int_a^b f - \int_a^b ap(f)\right| \leq \varepsilon|b-a|$$

y se logra un cálculo aproximado con m cifras decimales de precisión siempre que

$$\varepsilon < \frac{1}{|b-a|10^{m+1}} \tag{20}$$

Falta, por supuesto, que Σ conste de funciones "fáciles de integrar" para que todo esto sea práctico. Por ejemplo, esto sucede si Σ consta de todas las funciones de Φ que son lineales por tramos. Es muy fácil ver que el correspondiente problema de interpolación tiene solución única y entonces puede tomarse int como solución del problema de aproximación también, siempre y cuando la norma

$$h := \max_{0\leq i\leq N} |x_{i+1}-x_i|$$

sea suficientemente pequeña. En efecto, se demuestra que, en

este caso,

$$d(f,\text{int}(f)) \leq ch^2 \quad \text{si} \quad f \in C^2$$

por lo que basta tomar

$$0 < h < \sqrt{\frac{\varepsilon}{c}}$$

para lograr cualquier precisión —e incluso la condición (20)— si f es suficientemente suave. Este método de integración aproximada da lugar a la muy conocida fórmula de los trapecios, caso particular de (10).

Como un ejemplo final, consideremos el conjunto de soluciones de la ecuación diferencial

(21') $$y'' = F(x,y,y')$$

sobre [a,b] y sea ψ la solución que satisface las condiciones a la frontera

(21") $$y(a) = A \qquad y(b) = B$$

Para calcular ψ, podemos tomar como Φ al conjunto de todas las funciones continuamente diferenciables en [a,b], con segunda derivada continua en (a,b). Conviene tomar como distancia a

$$d(f,g) := d_0(f,g) + d_0(f',g'),$$

con d_0 como d en (19). Una clase $\Sigma \subset \Phi$ que resulta muy conveniente para el propósito de aproximar ψ es la siguiente:

Definición. a) Se dice que $\phi \in \Phi$ es un esplain cúbico si es un polinomio cúbico por secciones en $[a,b]$. En otras palabras, existe una partición $a =: x_0 < x_1 < \ldots < x_N < x_{N+1} := b$ de $[a,b]$ tal que

$$\phi(x) = b_{0i} + b_{1i}x + b_{2i}x^2 + b_{3i}x^3 \quad \text{si} \quad x_{i-1} \le x \le x_i,$$

pero

$$\phi^{(k)}(x_i^-) = \phi^{(k)}(x_i^+) \qquad k = 0,1,2$$

para $i = 1,\ldots,N$.

b) Se dice que ϕ es natural si $\phi''(a) = \phi''(b) = 0$.

Sea Σ la clase de todos los esplaines naturales sobre $[a,b]$. Es un resultado importante en Teoría de Aproximación [7] el que dice que, con estos elementos, el problema de interpolación tiene solución única; para obtenerla, basta resolver un sistema de ecuaciones lineales como (1), con matriz tridiagonal, lo cual simplifica las cosas. Se puede probar, además, que

$$d(f,\text{int}(f)) \le c(h^3 + h^4) \quad \text{para} \quad f \in C^4$$

de donde resulta inmediato que int también resuelve el problema de aproximación cuando se toman funciones suficientemente suaves.

En el artículo de Hugo Martínez se ilustran algunos métodos que se valen de esplaines para aproximar soluciones de

problemas del tipo de (21). En su artículo, Patricia Saavedra aplica estas ideas a la aproximación de soluciones de una ecuación en derivadas parciales, la de Burgers. Estos dos artículos se refieren al método del elemento finito, que junto con el de colocación y otros constituyen la clase de los métodos de residuos ponderados para discretizar ecuaciones diferenciales y otras. En su artículo, Erick Gamas aplica el método de colocación para resolver problemas de valores a la frontera que aparecen en ingeniería de reactores químicos. A su vez, Luis Mier y Terán aplica el mismo método a la solución de ecuaciones integrales no lineales de interés en Física Estadística.

4. Conclusiones.

El Análisis Numérico se ocupa de diseñar métodos efectivos para calcular números, vectores, funciones, funcionales, operadores, etc., tanto en forma directa como indirecta, es decir, cuando estos objetos están definidos mediante ecuaciones. Esta disciplina busca expresar dichos procedimientos en forma de algoritmos, los cuales se diseñan mediante una combinación de dos técnicas:

I. Reemplazar el problema a resolver por otro más sencillo, cuya solución aproxima a la del primero.

II. Reemplazar el problema a resolver por otro equivalente, el cual se resuelve en forma independiente.

Hemos visto también que estas técnicas se aplican de

manera reiterada, de tal manera a) de mejorar la aproximación (I), o bien b) lograr una formulación más conveniente (II). En el libro "Análisis Numérico" de esta misma serie hemos desarrollado estas ideas con amplitud.

Como se aprecia facilmente, el método I no es sino una relajación del II, que por ser más estricto no es muy frecuente que pueda aplicarse. Aquí hemos discutido ampliamente la aproximación como técnica para simplificar, según requiere el método I. Sin embargo, no es la única: algunas veces "complejo" significa "no lineal" y entonces "simplificar" significa "linealizar". El importantísimo método de Newton Raphson para calcular soluciones de ecuaciones no lineales (algebraicas o no) es solamente una aplicación reiterada de este método de simplificación.

También hemos identificado el problema de <u>calcular</u> con el de <u>aproximar</u>, y ha quedado claro que cuando de calcular se trate, invariablemente tendremos que valernos de un método numérico en algún punto del proceso. Además, observamos que para calcular hay que especificar a) la precisión requerida, b) el algoritmo de cálculo y c) el dispositivo para implantarlo.

Finalmente, hemos proporcionado un ambiente en el cual situar los restantes siete artículos que componen este volumen.

BIBLIOGRAFIA

[1] Bartle, R.G., "The elements of Real Analysis", John Wiley & Sons, Nueva York, 1964.

[2] Bender, C.M. y Orszag, S.A., "Advanced mathematical methods for scientists and engineers", McGraw Hill Book Co., Nueva York, 1978.

[3] Cárdenas, H., Luis, E., Raggi, F. y Tomás, F.J., "Algebra Superior", Ed. Trillas, Cd. de México, 1976.

[4] Courant, R. y Hilbert, D., "Methods of Mathematicl Physics" (Vol. I), Interscience Publishers, Inc., Nueva York, 1966.

[5] Davis, P.J., "Interpolation and approximation", Dover, Inc., Nueva York, 1975.

[6] Hernández, D.B. y Alvarez, L.J., "El método de Monte Carlo", Informe Monográfico, Departamento de Matemáticas UAM-Iztapalapa, México 1985.

[7] Prenter, P.M., "Splines and Variational Methods", Wiley-Interscience, Nueva York, 1975.

[8] Rudin, W., "Principles of Mathematical Analysis", McGraw Hill, Nueva York, 1976

[9] Rudin, W., "Real and Complex Analysis", McGraw Hill, Nueva York, 1964.

NOTAS SOBRE PROGRAMACION

Hans L. Fetter Nathansky*

Resumen

En estas notas presentamos a la programación como una disciplina creativa cuyo propósito es el de acercarnos cada vez más a la descripción detallada de los pasos a seguir en la solución de un problema por medio de una computadora.

Además se introducen unas herramientas muy poderosas, las estructuras de control básicas y la implementación de las mismas en un lenguaje muy conveniente: el pseudocódigo.

Se proporcionan algunos lineamientos generales que permiten, con relativa facilidad, escribir programas legibles y correctos.

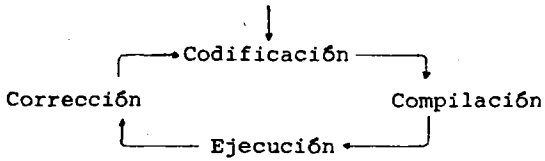
INTRODUCCION

Con cada vez mayor frecuencia, profesionistas de diferentes ramos (ingenieros, matemáticos, físicos, sociólogos, antropólogos, etc.) necesitan aprovechar las facilidades que les brindan las computadoras para resolver los problemas

* Departamento de Matemáticas, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana-Iztapalapa, Apartado Postal 55-534, 09340 México, D.F., México.

que les interesan: se puede tratar de un científico que desea resolver numéricamente ciertas ecuaciones diferenciales que modelan algún fenómeno físico, de un ingeniero que diseña el perfil aerodinámico para un nuevo avión, de un lingüista que elabora un diccionario para alguna lengua indígena o de un historiador que pretende desentrañar y esclarecer el ascenso al poder en el ámbito político mexicano (Smith, [1979]). Todas estas personas, en algún momento tomaron un curso introductorio de programación o bien estudiaron por iniciativa propia algún texto con un título como Introducción a la Programación con "el lenguaje X". Lamentablemente entonces, y todavía ahora, en dichos cursos o textos lo único que se les enseñaba era el repertorio de instrucciones, a veces muy extenso, del lenguaje X, pero no se les proporcionaba ningún tipo de lineamientos generales para desarrollar programas y mucho menos para que éstos resultaran además legibles, correctos y flexibles. Recordamos aquí las palabras del precursor de la programación estructurada (Dijkstra, [1972]): "En mi vida he visto varios cursos de programación que eran esencialmente como las clases de manejo usuales, en las que a uno le enseñan como maniobrar un automóvil en vez de cómo usar un automóvil para llegar al destino de uno". Y con la prisa o presión de comenzar con algo, la tendencia de muchos programadores novatos es la de codificar de inmediato algunas instrucciones en el lenguaje X. Este enfoque de atención y empleo de recursos a la etapa

de codificación —en detrimento de tres etapas previas indispensables: la de entendimiento o comprensión, la de análisis y la de programación del problema— dá como único resultado una dedicación de tiempo y de esfuerzos a un círculo vicioso, aparentemente sin fin, de:



En el mejor de los casos, al final de este proceso se obtienen programas mal diseñados (parchados), que funcionan correctamente, pero que resultan comprensibles únicamente para sus autores y eso sólo por un tiempo limitado. A menudo resultan programas que cumplen con sus funciones de una manera aproximada y deficiente o incluso, en algunos casos extremos, se obtienen programas que definitivamente no resuelven los problemas para los que fueron concebidos.

La programación no es una labor mecánica, sencilla, como lo es en general la codificación, sino que es un proceso intelectual bastante complejo. A este respecto escuchemos nuevamente la voz del maestro (Dijkstra, [1972]): "Ya deberíamos reconocer que la programación es un reto intelectual: el arte de programar es el arte de organizar la complejidad, de dominar la multiplicidad y de evitar el caos tanto como sea posible".

El propósito de estas notas es el de ayudar a acabar con el círculo vicioso antes mencionado y por ello ofrecemos a continuación algunos lineamientos generales que si bien no garantizan que el resultado de su aplicación sea un programa legible, correcto, flexible y simple, si por lo menos reducen los contratiempos y trastornos que se observan en el proceso de transición desde la concepción de un problema hasta su solución por medio de un programa para computadora.

FASES EN LA ELABORACION DE UN PROGRAMA

En esta sección queremos describir brevemente las etapas que se deben seguir para pasar de un problema informal a la solución del mismo mediante el uso de una computadora. Esto tiene gran interés, pues debe recordarse que la mayoría de los problemas se nos plantean de una manera vaga, ambigua, imprecisa; por ejemplo se nos dice: "Haga un programa para la nómina de esta empresa". Aunque esto puede parecer un problema rutinario para muchas personas, no podemos embarcarnos inmediatamente en el intento de elaborar un programa para esta tarea sin antes aclarar varios puntos: ¿cuáles y cuántos datos deberá manejar nuestro programa? ¡No es lo mismo hacer un programa para una empresa con 25 empleados que otro para una empresa con 25,000! ¿De qué forma vamos a tener acceso a la información necesaria como por ejemplo: identificación del trabajador, su salario, horas trabajadas, com-

pensaciones, deducciones, etc.? Y así como estas hay varias otras preguntas que debemos hacernos y cuyas respuestas debemos conocer antes de que intentemos resolver el problema que se nos ha planteado.

Otro ejemplo que nos permite observar más claramente las etapas previas a la programación y a la codificación nos lo proporciona el siguiente diálogo entre M (un matemático) y P (un programador).

M. "¿Puedes hacer un programa para comprobar la validez de la conjetura de Goldbach?"

P. "¿En qué consiste dicha conjetura?"

M. "Nos dice que todo número par mayor o igual que cuatro se puede expresar como la suma de dos números primos"

P. "Mmh"

P para convencerse de que ha entendido el problema, procede a verificar algunos casos particulares y se los muestra a M.

4 = 2+2, 6 = 3+3, 8 = 3+5, 10 = 3+7, 12 = 5+7, 14 = 3+11,...,
48 = 5+43, 50 = 3+47, 52 = 5+47, 54 = 7+47,...,21000 = 17+20983

M asiente y P con esta base, comienza a diseñar y a realizar una serie de experimentos mentales cuyo objetivo será el de acercarlo a la solución del problema, es decir, comienza a programar propiamente dicho. Pues como nos dice

Levine (Levine [1983]): "La finalidad de la programación es la de comprender con claridad el problema que va a ser resuelto o simulado por medio de la computadora y entender también con detalle cuál va a ser el procedimiento mediante el cual la máquina llegará a la solución deseada".

El (la) lector(a) interesado(a) puede intentar hacer un programa para verificar la conjetura anterior una vez que haya concluido con la lectura de las presentes notas.

El día en que nos podamos comunicar con una computadora de la misma manera que con un ser humano aun no ha llegado: una persona, con quien hablamos, nos entiende aun cuando no seamos muy claros y precisos al expresarnos e incluso, en ocasiones, nos sobreentiende. Una máquina no, ejecutará únicamente aquellas instrucciones que se le han proporcionado explícitamente y no tratará de adivinar cuáles fueron nuestras intenciones. Así pues, en la comunicación hombre-máquina la precisión debe ser la clave y si se desea obtener un programa en el lenguaje X que resulte claro, legible y correcto se deben seguir una serie de pasos que se describen a continuación de manera sucinta (Levine, [1983]):

1) Entender el problema.

Por obvio que parezca este paso, debemos de tomarlo en consideración, pues si no entendemos un problema, difícilmente seremos capaces de proponer un método para su solución. En el ejemplo anterior M tuvo que explicarle a P en qué con-

sistía la conjetura mencionada y en caso de que P no estuviera muy familiarizado con matemáticas M debería explicarle otros conceptos, como por ejemplo el de número primo.

Resulta difícil dar algun consejo específico para que este primer paso se lleve a cabo satisfactoriamente. Simplemente nos limitaremos a sugerir que el originador del problema y la persona que lo va a resolver se reúnan y discutan el problema con todo detalle. Como resultado de esta plática, la persona que originó el problema y que posiblemente no tenía conocimientos de computación, podría ahora sí redactar en español un enunciado del problema que tomase en cuenta, hasta cierto punto, tanto las capacidades como las limitaciones de las computadoras.

2) Analizar el problema.

Una vez entendido el problema, se deberá proceder a hacer un estudio del mismo con el objeto de proponer un método para su solución. Para ello resulta conveniente aislar los diferentes subproblemas del mismo y proponer algoritmos para la solución de cada uno de ellos. Por ejemplo, en el caso de la conjetura de Goldbach, un subproblema al que debemos dar solución es el siguiente: ¿Cómo determinamos si un cierto número es primo o no? Para cada uno de estos subproblemas debemos hacernos las siguientes preguntas: ¿Existen métodos conocidos ya sea en libros, revistas o en la programoteca, para resolverlos? ¿Cuáles de estos métodos debemos

emplear? Resulta que no siempre el primer método que viene a la mente es el mejor.

Asimismo se deberán estudiar cuidadosamente cuales son los datos que se van a manejar dentro del programa y cuales son las representaciones más adecuadas para los mismos.

3) Programar el método de solución propuesto.

La programación, a diferencia de la codificación que describiremos a continuación, es un proceso netamente creativo a través del cual, por medio de una serie de experimentos "mentales" que efectuamos sobre el problema que nos ocupa esperamos encontrar descripciones cada vez más detalladas de los pasos a seguir para resolverlo. Las descripciones iniciales se podrán hacer en cualquier lenguaje conveniente, pero conforme vamos cerrando la brecha entre lo que es comprensible para nosotros y lo que resulta inteligible para una máquina conviene emplear un lenguaje, conocido como pseudocódigo, que entre otras virtudes tiene la de ser legible para los humanos, amén de que una descripción expresada en dicho lenguaje se traduce a varios lenguajes de programación de uso común (PASCAL, ALGOL, FORTRAN, PL/1, BASIC, etc.) con suma facilidad, es decir, convierte el proceso de codificación en una tarea rutinaria.

Más adelante describiremos el pseudocódigo con más detalle y justificaremos su uso desde un punto de vista teórico.

4) Codificar el programa.

Usando la descripción escrita en pseudocódigo como guía, resulta una tarea muy sencilla y mecánica traducir dicha descripción a algún lenguaje de programación particular. Pero debemos enfatizar el hecho de que sólo ahora, (cuando contamos con una descripción en pseudocódigo que hemos podido probar mentalmente para convencernos de que en principio es correcta) y no antes podemos preocuparnos de introducir los detalles, a veces engorrosos, de un lenguaje de programación específico.

5) Ejecutar y ajustar el programa.

Una vez que hemos terminado de transcribir nuestra descripción en pseudocódigo a un lenguaje de programación tendremos que:

i) Perforar las instrucciones y los datos en tarjetas.

ii) Suministrar las tarjetas a la lectora de tarjetas.

iii) Esperar que el programa sea compilado, es decir, a que las instrucciones del lenguaje de programación sean traducidas al lenguaje de la máquina y también sean ejecutadas.

iv) Comprobar el correcto funcionamiento del programa.

Normalmente los pasos anteriores se repitan una y otra vez ya que suelen presentarse errores de diversa índole durante esta fase. Algunos de estos errores se pueden corregir

con relativa facilidad pues incluso nos son señalados explícitamente por medio del compilador a través de mensajes de diagnóstico apropiados como por ejemplo:

SIMBOLO NO PERMITIDO

FALTA PARENTESIS DERECHO

VARIABLE NO DEFINIDA

. . .

Otros errores, de tipo lógico, son por lo general mucho más difíciles de detectar y también de corregir. Los primeros son una señal de que el proceso de perforación o el de ensamblaje de las tarjetas se hicieron de una manera apresurada y/o descuidada mientras que los segundos apuntan hacia un análisis y/o programación deficientes.

Aprovechamos la oportunidad para recalcar que los esfuerzos del programador deben estar orientados hacia las tres primeras fases que hemos descrito, para evitar ajustes y depuramientos costosos e innecesarios.

6) Mantener el programa durante su vida útil.

Generalmente los programas no se usan una sola vez sino varias veces durante cierto período. Es por esto que debe pensarse en revisar y actualizar dichos programas periódicamente, ya que con el transcurso del tiempo pueden ocurrir cambios en la realidad que repercutan sobre los programas o también se pueden descubrir o redescubrir, métodos más efi-

cientes para resolver ciertos problemas o tareas.

LA PROGRAMACION ESTRUCTURADA

Para resolver un cierto problema se pueden proponer varios programas diferentes. Algunas de las razones para preferir uno de éstos sobre los demás son:

1) Es más fácil de leer.
2) Es más fácil de probar.
3) Es más fácil de modificar.
4) Requiere menos tiempo de ejecución.
5) Requiere menos espacio de memoria.

A continuación presentaremos a grandes rasgos una metodología que nos permita elaborar programas que reúnen las tres primeras cualidades. Estos programas no tienen porque ser totalmente ineficientes en cuanto a uso de tiempo y de espacio necesariamente, pero delegamos estos aspectos a una etapa posterior, pues antes de preocuparnos de la eficiencia de un programa debemos asegurarnos que funciona adecuadamente.

La razón por la cual deseamos escribir programas legibles, comprensibles y flexibles es porque éstos deberán ser leídos y entendidos no tanto por una computadora, sino por seres humanos que los actualizarán, corregirán y/o modificarán.

El término programación estructurada fue acuñado por Edsger W. Dijkstra (Dahl, O.J. et. al., [1972]), pero aún cuando el título de su artículo era "Notas sobre Programación Estructurada" no encontramos en él una definición de dicho concepto. De hecho, como veremos a continuación, se trata de un tópico muy polémico. A este respecto Van Tassel (Van Tassel, [1978]) nos dice que "la programación estructurada es un método que nos conduce a escribir mejores programas y cuyo objetivo es el de organizar y disciplinar los procesos de diseño y de codificación con el objeto de evitar la mayoría de los errores de lógica así como de facilitar la detección de aquellos que aún permanecen". Mientras que en Knuth (Knuth, [1974]) encontramos la siguiente cita atribuida a Mc Cracken: "Pocas personas osarían formular una definición de programación estructurada. De hecho, no está claro que exista una definición simple de dicho término aún". Lo cierto es que la programación estructurada no consiste, como comunmente se cree, en proscribir la proposición GO TO de los programas (un tema del cual hablaremos con más detalle más adelante) y sí comprende varios aspectos muy importantes como lo son:

i) el diseño arriba-abajo
ii) la modularidad
iii) el uso de estructuras de control básicas
iv) el empleo de un lenguaje conocido como pseudocódigo.

El diseño arriba-abajo.- Esta técnica nos es bastante conocida y consiste simplemente en identificar para un problema P dado una serie de subproblemas P1,P2,P3,... de tal forma que si logramos resolver estos últimos esto sea equivalente a haber resuelto el problema original P. Después este proceso se aplica a cada uno de los subproblemas y se repite tantas veces como sea necesario de manera que al final se tengan subproblemas lo suficientemente pequeños y simples que para su solución se disponga de algoritmos ya conocidos.

La modularidad.- La modularidad consiste en subdividir el problema en unidades lógicas (los módulos) autocontenidos y que pueden no ser directamente ejecutables, en el sentido de que para desempeñar sus funciones deben ser "llamados" por otros módulos. Si se ha efectuado un diseño arriba-abajo para un cierto problema se pueden considerar los distintos subproblemas como posibles candidatos para constituir módulos.

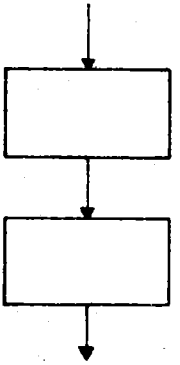
En el caso general se obtiene toda una jerarquía de módulos y es deseable que (Levine, [1983]):

a) Sean pequeños y sencillos,
b) oculten los detalles poco importantes a los módulos arriba de ellos en la jerarquía,
c) usen, a su vez, tantos módulos de menor jerarquía como sea necesario para cumplir con a),
d) sean legibles.

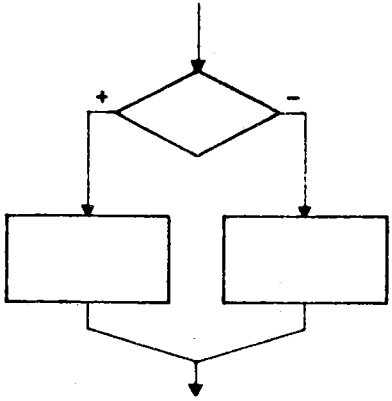
Estructuras de control básicas.- Lo que define en realidad a un programa son sus estructuras de control y de datos. Mediante las primeras dirigimos el flujo de acciones que se deberán efectuar sobre los datos, mismos que estarán organizados de maneras diversas que constituyen las estructuras de datos.

Una estructura de control básico con la que estamos muy familiarizados es la secuenciación y que se reduce a la observación de que en un programa las órdenes o instrucciones se ejecutan en el orden en que aparecen. Ahora bien, no todo algoritmo se puede expresar usando únicamente esta estructura de control básica. En ocasiones, durante la ejecución de un algoritmo debemos elegir entre dos alternativas y en base a esta decisión seguir uno u otro "camino". Esta estructura de control básica (que nos permite al tiempo de la ejecución, tomar decisiones sencillas guiados por alguna condición de tipo lógico), se conoce como selección. También surge a veces la necesidad de repetir una acción un cierto número de veces. Para ello podemos usar otra estructura de control básico, la iteración condicional y que, como su nombre lo indica, nos permite ejecutar una acción repetidas veces mientras se verifique una condición lógica. Nos basta con estas tres estructuras de control básico en el sentido que cualquier programa se puede expresar mediante ellas, como lo demostraron Böhm y Jacopini (Bohm, C. y G. Jacopini, [1966]) en lo que se conoce como el teorema de estructuras.

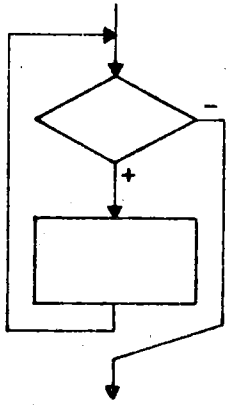
A continuación se muestra una representación gráfica de estas tres estructuras de control básicas mediante diagramas de flujo:



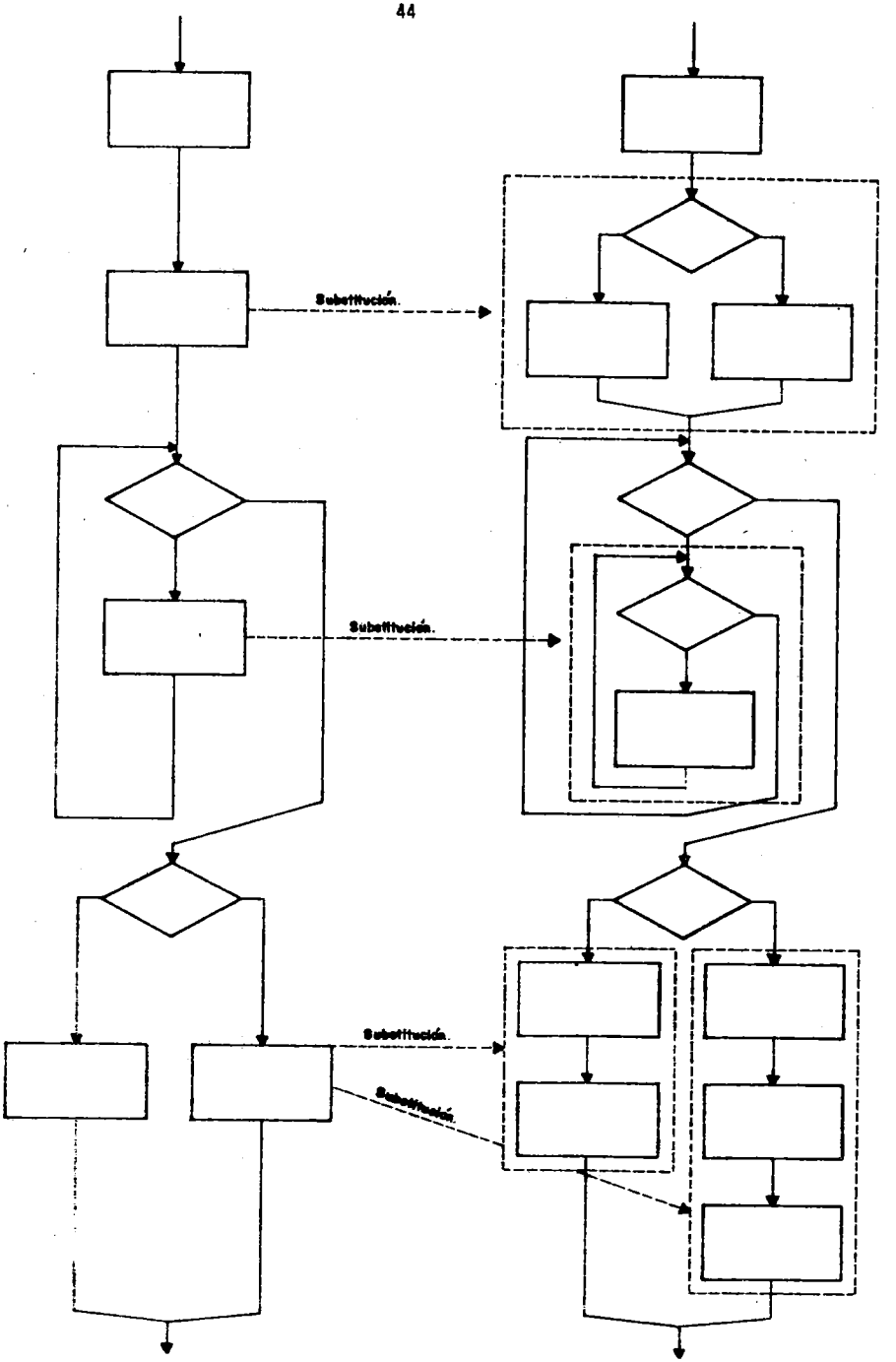
Secuenciación.



Selección.



Iteración Condicional.



Ejemplo 1.

Ejemplo 2.

En estos diagramas se aprecia la presencia de dos figuras geométricas: los rombos, que se utilizan como bloques de decisión y los rectángulos, que se emplean como bloques de procesamiento.

Un hecho que salta a la vista de la representación gráfica de las estructuras de control básicas y que resulta ser de suma importancia, es el siguiente: todas estas estructuras tienen un sólo punto de entrada y un sólo punto de salida. Esto nos permite ahora construir programas completos simplemente "pegando" entradas con salidas (ver ejemplo 1 a continuación) o incluso armar programas más complejos mediante un proceso de substitución, en el cual reemplazamos cualquier bloque de procesamiento por una combinación de las estructuras de control básicas (ver ejemplo 2).

Siguiendo la regla fundamental de composición (Levine, [1983]) "es posible combinar las estructuras de control de secuenciación, selección e iteración condicional" y en todos los casos el resultado será un programa bien formado (estructurado) con una sola entrada y una sola salida. Dichos programas bien formados poseen un alto grado de estructura, lo cual facilita leerlos de principio a fin. Ello trae como consecuencia que entender, probar y modificar dichos programas se simplifique considerablemente. En programas que no están bien estructurados los intentos de seguir la lógica se ven frustrados por constantes interrupciones, generalmente causadas por instrucciones del tipo GO TO. En estos programas

se ejecutan algunas instrucciones secuencialmente, luego, por medio de un GO TO ocurre una transferencia a otro lugar dentro del programa, ahí se ejecutan nuevamente en orden secuencial algunas instrucciones, después hay una transferencia a otro sitio en el programa y así sucesivamente. La persona que se enfrenta con este tipo de programas bien pronto, dentro de este constante ir y venir, pierde toda capacidad de asimilar el funcionamiento de dichos programas.

Es por esta razón que el GO TO es una estructura de control que por lo general se excluye de la programación estructurada" por su poder caótico y desestructurante sobre los programas" (Levine, [1983]). Sin embargo, la controversia sobre la eliminación total del GOTO es un asunto que aún no ha sido resuelto satisfactoriamente y que ha suscitado una serie de comentarios por parte de los conocedores del tema y que se reproducen a continuación:

"la calidad de un programador es inversamente proporcional al número de proposiciones GOTO en su programa". Dijkstra (Yourdon [1975]).

"el código de programadores establece: el uso de GOTO's puede ser nocivo para la salud de su programa". Van Tassel (Van Tassel, [1978]).

"nuestro conocimiento aún no ha alcanzado el punto en que los lenguajes del futuro deban eliminar el GOTO. Si el trabajo futuro indica que evitando los GOTO's podemos obtener algunas ganancias o ventajas significativas

> —tales como demostraciones rutinarias de que nuestros programas son correctos— entonces la decisión de mantener o retener la proposición GO TO debería reconsiderarse. Pero hasta entonces resulta sabio retenerla".
> Hopkins (Yourdon, [1975]).

El pseudocódigo.- Se pueden generar programas arbitrariamente grandes y complejos usando un conjunto de estructuras muy reducido: el de las estructuras de control básicas. Para expresar estos programas de manera clara, concisa y precisa nos valemos de un lenguaje denominado pseudocódigo. Destacan entre las principales virtudes de este lenguaje las siguientes:

a) legibilidad
b) neutralidad
c) completitud
d) semejanza con lenguajes de programación

Es un hecho que el pseudocódigo nos permite expresar la lógica de un programa de una manera muy clara y resulta mucho más conveniente que los diagramas de flujo.

Al usar pseudocódigo no necesitamos preocuparnos de respetar las reglas de algún lenguaje de programación particular ni tampoco de las peculiaridades de la máquina que vamos a emplear posteriormente. Es debido a esta característica fundamental —independencia tanto del lenguaje de programación como de la máquina que van a usarse— que decimos que el pseudocódigo es un lenguaje neutral.

La completitud del pseudocódigo es consecuencia del teorema de estructuras y se refiere a la capacidad que tiene este para expresar cualquier idea computacional.

La semejanza del pseudocódigo con varios lenguajes de programación de alto nivel como ALGOL, PASCAL, PL/1 entre otros es muy notoria y por consiguiente convierte la tarea de traducir un programa escrito en pseudocódigo en un programa escrito en alguno de estos lenguajes de alto nivel en un proceso mecánico y sumamente sencillo.

A continuación describimos los elementos fundametnales del pseudocódigo. Un programa en pseudocódigo es un conjunto de declaraciones de estructuras de datos, seguido de un conjunto de estructuras de control, todo encerrado entre las palabras programa (o procedimiento) y fin. Con respecto a las primeras nos limitaremos a decir que toda variable utilizada en un programa debe aparecer en una y sólo una declaración de estructura de datos. Veamos algunos ejemplos de declaraciones:

i) enterc I, J, K

(indica que las variables cuyos nombres son I, J y K se considerarán como del tipo entero dentro del programa).

ii) real PESO, ESTATURA, EDAD

(especifica que las variables cuyos nombres o identificadores son PESO, ESTATURA y EDAD son de tipo real).

iii) entero LISTA [1:10], POSICION [-10:10]

(esta declaración nos indica que LISTA es el nombre de un arreglo unidimensional o vector cuyos elementos LISTA [1], LISTA [2],..., LISTA [10] son variables de tipo entero, mientras que POSICION es el nombre de otro arreglo de variables enteras que consta de 21 elementos: POSICION [-10], POSICION [-9],...,POSICION [10]).

iv) real A[0:5, 0:4]

(en este caso A es el nombre de un arreglo bidimensional o matriz, cuyos 30 elementos A[0,0], A[0,1],..., A[0,4], A[1,0],..., A[5,4] son variables de tipo real cada una).

v) lógico LIBRE, ERROR

(LIBRE Y ERROR son los nombres de dos variables de tipo lógico, llamadas así porque sólo pueden tomar uno de los valores siguientes: falso o verdadero).

Las estructuras de control actúan sobre ciertas unidades básicas que denominamos enunciados y que representaremos de manera simbólica por la letra e (o e_i si hay necesidad de distinguir entre varios enunciados diferentes). Presentaremos a continuación algunos ejemplos de enunciados:

i) PI ← 3.14159

(este enunciado, que se conoce como proposición de asignación, nos indica que debemos asignar a la va-

riable cuyo nombre es PI el valor numérico 3.14159).

ii) I ← I + 1

(esta proposición aritmética se interpreta como sigue: evaluar la expresión que aparece a la derecha del símbolo ← y asignar el valor numérico resultante a la variable cuyo nombre aparece a la izquierda del símbolo ←).

iii) lee PESO, ESTATURA, EDAD

(esta proposición nos permite obtener valores numéricos para cada una de las variables, usando para ello un medio externo; por ejemplo: un teletipo, una pantalla de video o una lectora de tarjetas).

iv) escribe "RADIO =", R, "CIRCUNFERENCIA =", 2*PI*R

(permite al programa exhibir el valor de cada una de las variables y/o expresiones aritméticas y textos en un medio externo, como por ejemplo: en un teletipo, en una pantalla de video o en una impresora).

Pasemos ahora a la representación de las estructuras de control en este lenguaje:

i) Secuenciación: si e_1 y e_2 son dos enunciados entonces

e_1
e_2

o bien $e_1;e_2$

indican que se deben realizar o ejecutar los enuncia-

dos e_1 primero y luego e_2 en ese orden.

ii) Selección: si C es una condición lógica y e_3, e_4 son dos enunciados cualesquiera entonces

si C entonces e_3
otro e_4

es la proposición que nos dice lo siguiente: determínese el valor de verdad de la relación C. Si éste resulta verdadero, ejecútese el enunciado e_3 y en caso contrario, es decir, cuando éste es falso, ejecútese el enunciado e_4.

iii) Iteración condicional: si C es nuevamente alguna condición lógica y e_4 es cualquier enunciado entonces:

mientras (C) e_5

es la proposición cuyo efecto es el siguiente: evalúese la condición lógica C. Si el resultado es verdadero ejecútese el enunciado e_5 y luego repítase la operación desde el principio. Es claro que este ciclo repetitivo se romperá únicamente cuando la condición lógica C resulte tener el valor falso, por lo que el programador debe cerciorarse de que esto suceda en algún momento.

Observamos que en la descripción del pseudocódigo que hemos presentado hasta ahora aparecen ciertas palabras clave, que tienen funciones bien definidas. Estas palabras clave se

han subrayado para distinguirlas de las demás. Así por ejemplo: mientras, escribe, lee, entero, fin, etc. son palabras reservadas mientras que PESO, LISTA, ERROR, etc., no lo son.

También observamos que hasta ahora sólo hemos permitido que las estructuras de control actúen sobre enunciados individuales, pero ¿qué podemos hacer si deseamos que el entonces de una selección o el mientras de una iteracción condicional abarquen a todo un grupo o conjunto de enunciados? La respuesta es muy sencilla: encerramos la colección de enunciados entre las palabras clave comienza y termina, con lo cual hacemos que tengan la apariencia de un enunciado individual. Por ejemplo:

```
mientras (C)
    comienza
        e6
        e7
        e8
    termina
```

causa la repetida ejecución de los enunciados e_6, e_7 y e_8 mientras la condición lógica C sea verdadera.

Para incluir comentarios en nuestros programas simplemente colocamos el símbolo ! al inicio de los mismos. Ejemplos:

i) ! este procedimiento calcula el total, la media,
   ! la desviación estándar, el mínimo y el máximo
   ! para cada variable en un conjunto de observaciones.

ii) real A[1:100, 1:10] ! matriz de observaciones

Primer Ejemplo.

A continuación ilustraremos todo lo dicho anteriormente por medio de dos ejemplos bastante sencillos: uno de interés general y el otro de Análisis Numérico.

Para ello imaginémonos la siguiente situación:

Un alto jerarca (A.J.) del deporte se nos (N.) acerca y entonces se produce el siguiente diálogo:

A.J. "haga un programa para la competencia de clavados!"

N. "¿Qué es lo que este programa deberá hacer?"

A.J. "Deberá obtener la calificación de cada clavado."

N. "¿Cómo se obtiene la calificación de un clavado?"

A.J. "Cada juez, asignado a la competencia, otorga una puntuación entre cero y diez a un clavado. La puntuación más alta y la más baja se descartan y las restantes se promedian."

Posiblemente ya tengamos una idea vaga de qué deberá hacer nuestro programa, pero aún tenemos algunas dudas.

N. "¿Qué sucede si hay varias puntuaciones más altas (o más bajas) que son iguales, es decir, que están repetidas?"

A.J. "Unicamente se descarta una de ellas."

N. "¿Cuántos miembros hay en el jurado?"

A.J. "Normalmente hay entre ocho y doce. A tiempo de llevar-

se a cabo la competencia se le informará cual es el número exacto."

Probablemente ahora si ya entendimos el problema y por ello pasamos a la siguiente fase o sea la de análisis. ¿Cuáles son los datos del problema? Primero tenemos el número de jueces y segundo una lista de puntuaciones. ¿Cuáles son los resultados que se deberán obtener? La calificación del clavado calculada de acuerdo con la regla mencionada anteriormente. ¿Se conoce un método para la solución de este problema o bien podemos aislar subproblemas del mismo para los cuales sea factible encontrar dichos métodos? La respuesta a esta pregunta es si, como veremos enseguida, pero antes vamos a formular el problema con nuestras propias palabras.

Versión 0.

De una lista de puntuaciones entre cero y diez se deberán descartar la puntuación más alta y también la puntuación más baja y luego promediar las restantes.

Claramente reconocemos aquí el cuadro jerárquico de todos los problemas que se resuelven por medio de una computadora y que consiste en: entrada de datos, procesamiento de los mismos y salida de resultados.

Esto nos conduce inmediatamente a la siguiente descripción de los pasos a seguir para resolver el problema que nos ocupa, incorporando ya algunos elementos del pseudocódigo:

Versión 1.

```
Programa
! PRIMERA DESCRIPCION DEL PROGRAMA DE CALIFICACION DE
  CLAVADOS
leer los datos
procesar los datos
reportar los resultados
fin
```

¿De qué forma llevamos a cabo cada uno de los tres pasos en la descripción anterior? De manera aproximada podemos decir que

"leer los datos" se reduce a leer el número de jueces y la lista de puntuaciones,

"procesar los datos" consiste en determinar la puntuación máxima y la puntuación total, así como también promediar las puntuaciones adecuadamente,

"reportar los resultados" no es más que escribir el valor del promedio.

Es aquí cuando ya empezamos a sospechar que sí contamos con métodos para resolver los diferentes subproblemas en que hemos subdividido nuestro problema. Más adelante describiremos estos métodos con cada vez más precisión. Por ahora nos limitaremos a presentar la siguiente descripción del programa.

Versión 2.

Programa

! SEGUNDA DESCRIPCION DEL PROGRAMA DE CALIFICACION DE CLAVADOS

lee número de jueces

lee lista de puntuaciones

hallar puntuación mínima, puntuación máxima y puntuación total

calcular calificación promediando puntuaciones adecuadamente

escribe calificación

fin.

En la mayoría de los programas se debe verificar la validez de los datos de entrada y el presente programa no es la excepción: recordemos, por ejemplo, el hecho de que el número de jueces no es arbitrario, sino que está restringido a ciertos valores. Es por esto que después de leer el número de jueces debemos proceder a verificar si este número está o no dentro del rango permitido.

También hacemos la observación de que resulta conveniente mandar escribir los datos que previamente se leyeron para detectar posibles errores si el volúmen de aquellos no es demasiado grande.

Para determinar la puntuación mínima, debemos primero inicializar ésta de alguna manera y luego, conforme vamos

observando las puntuaciones de una en una, actualizarla debidamente. De manera similar se pueden obtener la puntuación máxima y la puntuación total. Podemos aprovechar el hecho de que vamos a analizar las puntuaciones de una en una para verificar si éstas se encuentran dentro del rango permitido.

Para calcular la calificación nos basta con restarle a la puntuación total la suma de la puntuación mínima y de la puntuación máxima y promediar adecuadamente.

Todo esto y algunas cosas más los encontramos sintetizados en la siguiente versión:

Versión 3.

```
Programa
: TERCERA DESCRIPCION DEL PROGRAMA DE CALIFICACION DE
  CLAVADOS
lee número de jueces
si número de jueces está fuera del rango permitido
   entonces reporta error y termina ejecución
lee lista de puntuaciones
inicializa puntuación mínima, puntuación máxima y pun-
   tuación total
observa el primer elemento de la lista de puntuaciones
   : considérala como
   : la puntuación actual
mientras (no se agote la lista de puntuaciones
comienza
```

si puntuación actual está fuera del rango permitido
entonces reporta error y termina ejecución

actualiza puntuación mínima, puntuación máxima y puntuación total

observa el siguiente elemento de la lista de puntuaciones

termina

calcula calificación, restándole a la puntuación total la puntuación mínima y la puntuación máxima y promediando adecuadamente

escribe calificación

fin.

Esperamos que se comprenda cuál es el funcionamiento de este programa, aún cuando falten muchos detalles por especificar. Entre otras cosas observamos que en este programa hay de hecho tres salidas: una, la salida normal al final del programa y otras dos cuando detectemos algún error entre nuestros datos, es decir, si el número de jueces está fuera del rango permitido o bien si sucede lo mismo para alguna puntuación.

Para remediar esta situación proponemos la siguiente solución:

1. Cambiar: si número de jueces está fuera del rango permitido entonces reporta error y termina

ejecución

por: si número de jueces está fuera del rango permitido entonces reporta error

otro "procesa las puntuaciones"

2. Antes de entrar al ciclo para procesar las puntuaciones levantar una bandera de éxito, indicando con ello que hasta ese momento no se ha detectado error alguno en las puntuaciones y además

cambiar: si puntuación actual está fuera del rango permitido entonces reporta error y termina ejecución

por: si puntuación actual está fuera del rango permitido entonces baja bandera de éxito

otro ...

Al salir del ciclo se deberá observar la bandera de éxito y dependiendo de su estado proceder con el cálculo de la calificación o reportar el hecho de que hubo alguna puntuación fuera del rango permitido.

Esto dá como resultado al siguiente descripción:

Versión 4.

Programa

! CUARTA DESCRIPCION DEL PROGRAMA DE CALIFICACIONES DE CLAVADOS

lee número de jueces

```
si número de jueces está fuera del rango permitido
   entonces reporta error en el número de jueces
   otro comienza
        lee lista de puntuaciones
        escribe lista de puntuaciones
        inicializa puntuación mínima, puntuación máxima
           y puntuación total
        levanta bandera de éxito
        observa el primer elemento de la lista de pun-
           tuaciones !considéralo como la puntuación
                     !actual
        mientras (no se agote la lista de puntuaciones)
        comienza
           si puntuación actual está fuera del rango
              permitido entonces baja bandera de éxito
                         otro
                              actualiza puntuación mínima,
                              puntuación máxima y puntua-
                              ción total
                         observa el siguiente elemento
                         de la lista de puntuaciones
        termina
        si bandera de éxito está levantada
           entonces comienza
                    calcula calificación, restándole a
                    la puntuación total la puntuación
```

```
                    mínima y la puntuación máxima y
                    promediando adecuadamente.
                    escribe calificación
                termina
            otro reporta error en puntuaciones
        termina
fin.
```

Esta descripción del algoritmo padece de algunas inconveniencias ligeras: al detectar una puntuación fuera del rango permitido se siguen procesando las demás puntuaciones, lo cual a estas alturas ya no es necesario. Tampoco se proporciona información específica sobre cuál puntuación causó que se bajara la bandera de éxito.

Para evitar el procesamiento innecesario de las puntuaciones una vez que se encontró un error en alguna de ellas, proponemos integrar al mientras una prueba sobre el estado de la bandera de éxito. Como veremos en versiones posteriores del programa, esto a su vez también nos permitirá indicar cuál fue la primera puntuación que se encontró fuera del rango permitido.

Así pues, nuestra siguiente descripción queda como sigue:

Versión 5.

```
Programa
! QUINTA DESCRIPCION DEL PROGRAMA DE CALIFICACIONES DE
  CLAVADOS
lee número de jueces
si número de jueces está fuera del rango permitido
   entonces reporta error en el número de jueces
   otro comienza
         lee lista de puntuaciones
         escribe lista de puntuaciones
         inicializa puntuación mínima, puntuación máxi-
            ma y puntuación total
         levanta bandera de éxito
         observa el primer elemento de la lista de pun-
            tuaciones
                  !considéralo como
                  !la puntuación actual
         mientras (no se agote la lista de puntuaciones
                  y bandera de éxito esté levantada)
         comienza
            si puntuación actual está fuera del rango
                  permitido
               entonces baja bandera de éxito
               otro actualiza puntuación mínima, pun-
                  tuación máxima y puntuación total
            observa el siguiente elemento de la lista
```

```
                    de puntuaciones
                termina
                si bandera de éxito está levantada
                   entonces comienza
                            calcula calificación, restándole
                            a la puntuación total la puntua-
                            ción mínima y la puntuación máxi-
                            ma y promediando adecuadamente
                            escribe calificación
                            termina
                   otro reporta la puntuación que causó se
                        bajara la bandera de éxito
                termina
    fin.
```

Nosotros entendemos intuitivamente operaciones o conceptos como:

```
levanta bandera de éxito
inicializa puntuación mínima, puntuación
           máxima y puntuación total
observa el primer elemento de la lista de
           puntuaciones
lista de puntuaciones
bandera de éxito
etc.
```

pero difícilmente vamos a encontrar un lenguaje de programa-

ción que tenga instrucciones o descripciones de esta naturaleza entre su repertorio. Es por esto que ahora necesitamos ocuparnos de las estructuras de datos y de las operaciones que se pueden efectuar sobre las mismas (comparaciones, asignación de valores, referencias a elementos de las estructuras). En efecto, es precisamente a través de éstas que podemos (Levine, [1983]) "descomponer la carga semántica de un programa en elementos más cercanos a la naturaleza operacional de una computadora". Por ejemplo, para representar la lista de puntuaciones vamos a emplear un arreglo unidimensional o vector:

real puntuación [1:12]

Observar un elemento de la lista de puntuaciones se puede realizar mediante el uso de un índice, así

puntuación [i]

hace referencia a la i-ésima componente del vector puntuación.

La bandera de éxito no será más que una variable de tipo lógico

lógico nohayerror

y las operaciones de levantar o bajar esta bandera se realizarán asignándole los valores verdadero y falso respectiva-

mente.

Para determinar la puntuación mínima (pmínima) podemos proceder de la manera siguiente:

Primero le asignamos a pmínima el mayor valor permitido para las puntuaciones (10) —inicialización— y una vez procesadas las i-1 primeras puntuaciones se obtiene el valor de pmínima como el mínimo entre la i-ésima puntuación y el mínimo obtenido para las i-1 puntuaciones anteriores (i = 1,2, ...,numjueces) —actualización. De manera análoga se calculan pmáxima (la puntuación máxima) y ptotal (la puntuación total).

Utilizando nombres mnemónicos para nuestras variables, es decir, nombres que nos dan una idea clara del propósito de estas variables dentro del programa obtenemos ahora la siguiente descripción de nuestro algoritmo:

Versión 6.

```
Programa
! SEXTA DESCRIPCION DEL PROGRAMA DE CALIFICACIONES DE
  CLAVADOS
! Declaraciones de las estructuras de datos usadas
real puntuación [1:12]
real pmínima, pmáxima, ptotal, calificacn
entero numjueces, i
lógico nohayerror
! Proposiciones del programa
lee numjueces
```

```
si numjueces < 8 o numjueces > 12
   entonces escribe "el número de jueces (",numjueces,")
                    no es aceptable"
   otro comienza
          lee (puntuación [i], i ← 1,...,numjueces)
          escribe "puntuaciones otorgadas", (puntuación
                  [i], i ← 1,...,numjueces)
          pmínima ← 10.0
          pmáxima ←  0.0
          ptotal  ←  0.0
          nohayerror ← verdadero
          i ← 1
          mientras (i ≤ numjueces y nohayerror)
          comienza
             si puntuación [i] < 0.0 ó puntuación
                           [i] > 10.0
                entonces nohayerror ← falso
                otro comienza
                        pmínima ← mín (puntuación [i],
                                       pmínima)
                        pmáxima ← máx (puntuación [i],
                                       pmáxima)
                        ptotal ← ptotal + puntuacion [i]
                     termina
                i ← i + 1
             termina
```

```
si nohayerror entonces comienza
                        calificacn←(ptotal-
                        (pmínima+pmáxima))
                        /(numjueces - 2)
                        escribe "la califi-
                        cación del clavado
                        es", calificacn
                    termina
                otro escribe "la calificación
                        otorgada por el juez",
                        i - 1, "está fuera del
                                rango"
            termina
fin.
```

Sólo esperamos que, después de cinco descripciones previas del mismo algoritmo, la versión final se explique por si misma y además se reconozca la facilidad con que se codificaría ahora este algoritmo en lenguajes como ALGOL, PASCAL o FOREST.

Asimismo invitamos a los lectores a que intenten resolver una pequeña variante del problema anterior: de una colección de N puntuaciones se deberán descartar las M puntuaciones más altas y también las M más bajas y luego promediar las restantes. Este problema es bastante más complicado que el anterior y no requiere que ordenemos todas las puntuaciones

previamente.

Segundo Ejemplo.

El problema, en este caso, es de Análisis Numérico, y consiste en determinar factores cuadráticos de un polinomio $p_N(x)$ de grado N. Es decir, se trata de encontrar dos números reales u y v tales que $x^2 - ux - v$ sea un factor exacto del polinomio dado $p_N(x)$. Se debe de cumplir, pues, que

$$p_N(x) = (x^2 - ux - v)\cdot q_{N-2}(x)$$

donde $q_{N-2}(x)$ representa un polinomio de grado N-2.

Lo más indicado para entender y analizar este problema es consultar la literatura: Véanse por ejemplo, (Henrici, [1964]) y (Isaacson-Keller, [1966]). Supondremos que el lector/la lectora ya llevó a cabo esta labor y nos limitaremos a presentar el método de Bairstow, como se describe en estos textos y luego a expresarlo en nuestro lenguaje: pseudocódigo.

A saber, el algoritmo es el siguiente:

Versión 1.

Dado el polinomio

$$p_N(x) = a_0x^N + a_1x^{N-1} + a_2x^{N-2} + \ldots + a_N$$

y un factor cuadrático inicial

$$x^2 - u_0x - v_0$$

generar una sucesión $\{x^2 - u_nx - v_n\}$ de factores cuadráticos como sigue:

Para n = 0,1,2,.... (hasta obtener un resultado satisfactorio) genérese la sucesión $\{b_i\} = \{b_i^n\}$

a partir de: $b_{-2} = b_{-1} = 0$ y

$$b_i = a_i + u_n b_{i-1} + v_n b_{i-2} \quad i = 0,1,\ldots,N$$

genérese la sucesión $\{c_i\} = \{c_i^n\}$

a partir de: $c_{-2} = c_{-1} = 0$ y

$$c_i = b_i + u_n c_{i-1} + v_n c_{i-2} \quad i = 0,1,\ldots,N-1$$

resuélvase el sistema de ecuaciones simultáneas:

$$c_{N-2}\delta + c_{N-3}\varepsilon = -b_{N-1}$$

$$c_{N-1}\delta + c_{N-2}\varepsilon = -b_N$$

hágase

$$u_{n+1} = u_n + \delta$$

$$v_{n+1} = v_n + \varepsilon$$

En realidad el polinomio $p_N(x)$ queda totalmente determinado por sus coeficientes $a_0, a_1, \ldots, a_N$, lo mismo que el fac-

tor cuadrático inicial $x^2 - u_0x - v_0$ queda determinado por los coeficientes u_0, v_0. Con esto en mente proponemos ahora la siguiente descripción del algoritmo de Bairstow:

Versión 2.

```
Procedimiento Bairstow
! SEGUNDA DESCRIPCION DEL METODO DE BAIRSTOW
! Supone que N,a0,a1,...,aN  y  u0, v0  son conocidos
! u0, v0  es la aproximación actual de la solución
          exacta
mientras (la aproximación actual de la solución exacta
         u, v no se considere suficientemente buena)
comienza
   genera sucesión {bi}
   genera sucesión {ci}
   resuelve sistema de ecuaciones lineales
   actualiza aproximación para u, v  y considérala como
         la actual
termina
fin.
```

Aunque podríamos hacerlo, no presentamos de una vez la versión final de este pequeño procedimiento porque (Denning, |1974|): "No es suficiente mostrar el producto final y esperar que el lector perciba su estructura por inspección o incluso por profunda meditación. En su lugar, el lector debe

ser capaz de ver al menos parte del proceso mental del programador, comenzando por la versión original (muy vaga) y procediendo hacia el producto final a través de una secuencia claramente presentada de transformaciones claras y refinamientos".

De hecho, lo único que hemos hecho al pasar de la versión 1 a la 2, es indicar de qué manera se van a proporcionar ciertos datos del problema, el polinomio $p_N(x)$ y el factor cuadrático inicial $x^2 - u_0x - v_0$, así como el traducir el Para n = 0,1,2,... (hasta obtener un resultado satisfactorio) en una proposición válida de pseudocódigo.

Generar las sucesiones $\{b_i\}$ y $\{c_i\}$ no presenta mayor dificultad, pues se trata de ecuaciones en diferencias con valores iniciales, que se pueden resolver explícitamente por sustitución.

Para resolver el sistema de ecuaciones lineales simultáneas resultante recomendamos, por tratarse de un sistema de 2 x 2, usar la regla de Cramer y que en este caso nos dice:

$$\delta = \frac{\begin{vmatrix} -b_{N-1} & c_{N-3} \\ -b_N & c_{N-2} \end{vmatrix}}{\begin{vmatrix} c_{N-2} & c_{N-3} \\ c_{N-1} & c_{N-2} \end{vmatrix}} \qquad \varepsilon = \frac{\begin{vmatrix} c_{N-2} & -b_{N-1} \\ c_{N-1} & -b_N \end{vmatrix}}{\begin{vmatrix} c_{N-2} & c_{N-3} \\ c_{N-1} & c_{N-2} \end{vmatrix}}$$

Queremos mencionar que en la expresión para ε en el

libro de Henrici hay un pequeño error. Esto nos muestra que no debemos depositar una confianza ciega y absoluta en lo que alguien publica. A este respecto queremos recomendar la lectura de un breve artículo por P.J. Davis (Davis, [1972]), que discute la frecuencia de errores en publicaciones sobre matemáticas.

Observamos que en la primera versión del algoritmo de Bairstow empleamos subíndices para diferenciar las aproximaciones sucesivas que se van obteniendo. En realidad esto no es necesario, ya que una vez conocidos los valores de u_{n+1} y v_{n+1} no requerimos de los anteriores: $u_n, v_n, \ldots, u_1, v_1, u_0, v_0$.

El proceso de actualizar la aproximación para u, v y considerarla como la actual es inmediato.

Finalmente, y antes de presentar la siguiente versión del procedimiento, sólo queremos mencionar que para generar las sucesiones $\{b_i\}$ y $\{c_i\}$ no se tiene que obtener una después de la otra, sino que se pueden generar en paralelo como lo hacemos a continuación:

Versión 3.

```
Procedimiento Bairstow
! TERCERA DESCRIPCION DEL METODO DE BAIRSTOW
! Supone que N,a0,a1,...,aN y u,v son conocidos
! u,v es la aproximación actual de la solución exacta
mientras (la aproximación actual de la solución exacta
          no se considere suficientemente buena)
```

comienza

$b_{-2} \leftarrow 0.0$

$b_{-1} \leftarrow 0.0$

$c_{-2} \leftarrow 0.0$

$c_{-1} \leftarrow 0.0$

$i \leftarrow 0$

mientras $(i \leq N)$ comienza

$b_i \leftarrow a_i + ub_{i-1} + vb_{i-2}$

$c_i \leftarrow b_i + uc_{i-1} + vc_{i-2}$

$i \leftarrow i + 1$

termina

$det \leftarrow c_{N-2}^2 - c_{N-1} \cdot c_{N-3}$

$\delta \leftarrow (b_N \cdot c_{N-3} - b_{N-1} \cdot c_{N-2})/det$

$\varepsilon \leftarrow (b_{N-1} c_{N-1} - b_N \cdot c_{N-2})/det$

$u \leftarrow u + \delta$

$v \leftarrow v + \varepsilon$

termina

fin.

Aún debemos limar varias asperezas. Por ejemplo, debemos hacer más operacional nuestro criterio de terminación. Algunas posibilidades que se presentan son:

terminar si a) $\max(|\delta|, |\varepsilon|) < tol_1$

o si b) $\max(|\delta|,|\varepsilon|)/\max(|u|,|v|) < tol_2$

o si c) el número de iteraciones es mayor que $N_{máx}$ para cierto número de iteraciones $N_{máx}$ dado

Aquí, tol_1 y tol_2 dados son tolerancias del error absoluto y relativo respectivamente.

Además deseamos incluir ahora las estructuras de datos sugeridas por el propio algoritmo.

El algoritmo, entonces, queda de la manera siguiente:

Versión 4.

```
Procedimiento Bairstow (a,N,u,v,N_máx,tol_1,tol_2)
! CUARTA DESCRIPCION DEL METODO DE BAIRSTOW
! Declaraciones de las estructuras de datos usadas
real a[0:N], b[-2:N], c[-2:N]
real u, v, tol_1, tol_2, errorabs, errorrel, det, δ, ε
entero i, n, N_max, N
escribe "polinomio de grado", N, "con coeficientes",
        (a[i],i ← 0,1,...,N)
n ← 0
errorabs ← 1.0
errorrel ← 1.0
escribe "iteración", "  u  ", "  v  ", "error abs",
        "error rel", "tol error abs", "tol error rel"
escribe  n , u , v , errorabs, errorrel, tol_1, tol_2
```

```
mientras (n ≤ N_max y errorabs ≥ tol_1 y errorrel ≥ tol_2)
   comienza
            b[-2] ← b[-1] ← c[-2] ← c[-1] ← 0.0
            i ← 0
      mientras (i ≤ N) comienza

                        b[i] ← a[i] + u·b[i-1] + v·b[i-2]

                        c[i] ← b[i] + u·c[i-1] + v·c[i-2]

                        i ← i + 1
                     termina
      det ← c[N-2]·c[N-2] - c[N-1]·c[N-3]
        δ ← (b[N]·c[N-3] - b[N-1]·c[N-2])/det
        ε ← (b[N-1]·c[N-1] - b[N]·c[N-2])/det
        u ← u + δ
        v ← v + ε
      errorabs ← máx(|δ|,|ε|)
      errorrel ← errorabs/máx(|u|,|v|)
      n ← n + 1
      escribe   n , u , v , errorabs, errorrel
   termina
fin.
```

Invitamos al lector/a a que elaboren un programa en pseudocódigo, para encontrar el valor propio dominante de una matriz simétrica A y un vector propio asociado de norma

uno por el método de las potencias, cuyo algoritmo describimos a continuación (Burden et. al. [1978] y Hernández [1983]):

Dado un vector $\tilde{y}^{(0)} \neq \tilde{0}$:

1. Hágase $\tilde{x}^{(0)} = \tilde{y}^{(0)} / \|y^{(0)}\|_2$

2. Hágase $m = 1$

3. Calcúlese $\tilde{y}^{(m)} = A\tilde{x}^{(m-1)}$

4. Si $\tilde{y}^{(m)} = \tilde{0}$, entonces 0 es un valor propio de A con vector propio $\tilde{x}^{(m-1)}$, escójase otro vector $\tilde{y}^{(0)}$ y regrésese al paso 1.

5. Hágase $\mu^{(m)} = (\tilde{x}^{(m-1)})^t \tilde{y}^{(m)}$

6. Hágase $\tilde{x}^{(m)} = \tilde{y}^{(m)} / |\tilde{y}^{(m)}|_2$

7. Si $\mu^{(m)}$ es una aproximación suficientemente buena del valor propio dominante buscado váyase al paso 10.

8. Increméntese m en uno

9. Regrésese al paso 3.

10. El procedimiento ha concluído: $\mu^{(m)}$ es una buena aproximación al valor propio dominante de A y $\tilde{x}^{(m)}$ es un vector con $|\tilde{x}^{(m)}|_2 = 1$ que aproxima al vector propio asociado.

CONCLUSIONES

En estas notas hemos examinado algunas técnicas que nos conducen a obtener programas legibles. Existen otras y no las hemos ignorado por completo, pero si hemos menospreciado su importancia, a saber, podemos mencionar:

a) el uso de comentarios apropiados
b) la elección de nombres adecuados para variables, funciones, procedimientos, etc.
c) el estilo, la claridad y la creatividad del programador
d) la presentación del texto del programa (por ejemplo, el uso de sangrado para exhibir la estructura del programa).

Tampoco nos hemos preocupado de la eficiencia de nuestros programas. La eficiencia tiene por objeto reducir algún costo ya sea de espacio o de tiempo o de ambos. Pero esta es una etapa necesariamente posterior a la de programación, ya que mientras un programa no funcione, no importa cuan eficiente sea.

También hemos descuidado otro aspecto de los programas que es su tamaño: no es lo mismo elaborar un programa que consta de veinte líneas que otro que consta de cien. Si empleamos un tiempo t para desarrollar el primer programa bien podría ser que para elaborar el segundo no vamos a emplear un tiempo de 5t, sino más bien un tiempo de $25t^2$!

Finalmente, esperamos que el lector/la lectora ya no tenga la idea muy equivocada de que la programación estructurada es un proceso de escribir programas de los cuales posteriormente se eliminan todas las ocurrencias de la proposición GO TO. Al contrario, las metas de la programación estructurada consisten en diseñar métodos ordenados y eficientes para desarrollar programas legibles y correctos, así como de identificar y explicar herramientas y técnicas para resolver problemas de programación y de como pensar claramente al programar (Gries, [1978]).

Concluímos con la tesis de Benjamin Lee Whorf (Gries, [1978]) basada en sus estudios sobre las lenguas europeas y las indígenas americanas, pero que se aplica de igual manera a los lenguajes de programación

"El lenguaje modela al pensamiento y a la cultura de quienes lo usan".

BIBLIOGRAFIA

1. Böhm, C. & G. Jacopini, "Flow Diagrams, Turing Machines and Languajes with only two Formulation Rules", Communications of the ACM, Mayo 1966, Páginas 366-371.
2. Burden, Richard L., J. Douglas Faires & Albert C. Reynolds, "Numerical Analysis", Boston, Mass.: Prindle, Weber F. Schmidt, 1978.
3. Davis, P.J., "Fidelity in Mathematical Discourse: Is one and one Really two?", American Mathematical Monthly, Marzo 1972.
4. Denning, Peter J., "Guest Editor's Overview...", ACM Computing Surreys, Vol. 6, No. 4, Dic. 1974, págs. 209-211.
5. Dijkstra, Edsger W., "Notes on structured Programming" en Dahl, O.J., Edsger W. Dijkstra & C.A.R. Hoare, Structured Programming, Nueva York: Academic Press 1972.
6. Gries, David, "On Structured Programming" en Gries, David Ed., Programming Methodology. A collection of articles by Members of IFIP WG 2.3. Nueva York: Springer Verlag 1978.
7. Henrici, Peter, "Elements of Numerical Analysis", Nueva York: John Wiley & Sons 1964.
8. Hernández, Diego B., "Análisis Numérico: Discretización y Algoritmica", Notas del 3er. Coloquio del Departamento de Matemáticas (La Trinidad-Tlaxcala), CIEA-IPN, México, D.F. 1983.

9. Isaacson, Eugene & Herbert B. Keller, "Analysis of Numerical Methods", Nueva York: John Wiley & Sons 1966.

10. Knuth, Donald E., "Structured Programming with GO TO Statements", ACM Computing Surveys, Vol. 6, No. 4, Dic. 1974, págs. 261-302.

11. Levine, Guillermo, "Manual del usuario del sistema Forest", UAM-Iztapalapa, México, D.F. 1980.

12. Levine, Guillermo, "Introducción a la Computación y a la Programación Estructurada", Reporte Interno, UAM-Iztapalapa, México, D.F. 1983.

13. Smith, Peter, H., "Labyrinths of Power". Political Recruitment in Twentieth-Century Mexico. Princeton, New Jersey: Princeton University Press 1979.

14. Van Tassel, Dennie, "Program Style, Design, Efficiency, Debugging and Testing. Segunda Edición. Englewood Cliffs, N.J.: Prentice Hall 1978.

15. Yourdon, Edward, "Techniques of Program Structure and Design", Englewood Cliffs, N.J.: Prentice Hall 1975.

APROXIMACION DE FUNCIONES MEDIANTE MAQUINAS DIGITALES: EL ALGORITMO CORDIC

Carlos Velarde*

Resumen

En este trabajo se presenta una de las formas utilizadas por las máquinas digitales para aproximar funciones elementales, incluyendo sen, cos, arctan, senh, cosh, arctanh, exp, ln, multiplicación y división.

El algoritmo de aproximación es común para todas estan funciones, y de tal sencillez que se han construido circuitos electrónicos ("hardware") para él, optimizando así el tiempo de evaluación. Estos circuitos son usados en algunas calculadoras de bolsillo y en procesadores de punto flotante [1,3]. En la primera parte se describe el algoritmo y se exponen sus fundamentos matemáticos. En la segunda sección damos una comparación del algotimo con el método de Euler.

1. El Algoritmo

La clave del algoritmo consiste en la aplicación reiterada de las fórmulas

* Centro de Investigación en Matemáticas (CIMAT), Apdo. Postal 402, Guanajuato, Gto. C.P. 46000 e Instituto de Investigación en Matemáticas Aplicadas y en Sistemas (UNAM), Cd. Universitaria, C.P. 04510, México, D.F.

$$
(1)\qquad
\begin{aligned}
x_{k+1} &= x_k - m\delta_k y_k 2^{-k} \\
y_{k+1} &= y_k + \delta_k x_k 2^{-k} \qquad \delta_k = \pm 1 \\
z_{k+1} &= z_k - \delta_k \varepsilon_k
\end{aligned}
$$

a partir de valores iniciales para m (1,0 ó -1), x_0, y_0, z_0, usando una lista de constantes $\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_n$, y escogiendo δ_k mediante la simple comparación de y_k o z_k con 0.

Observando que en notación binaria la multiplicación por 2^{-k} se puede realizar mediante un corrimiento del punto binario, nos damos cuenta que la evaluación de las fórmulas anteriores sólo requiere de las operaciones suma, resta, corrimiento, cambio de signo, comparaciones sencillas y acceso a cada constante ε_k. Para estas operaciones no es difícil diseñar circuitos que las realicen electrónicamente, y con éstos se han fabricado microprocesadores especiales para aritmética de punto flotante [1].

Veamos ahora una justificación matemática del algoritmo. Para ello conviene tener presente que las máquinas digitales, al evaluar una función numérica f en un número t, en muchas ocasiones lo que calculas es una proximación $f_M(t_M)$ del número deseado $f(t)$ (Figura 1). Esto es inevitable por la sencilla razón de que el conjunto de números representables en una máquina es finito, y t o $f(t)$ pueden no pertenecer a él.

En los términos anteriores, el algoritmo CORDIC construye una aproximación t_M, garantizando que el error α es menor

que ε_n, a la vez que genera el número $f_M(t_M)$, con todas sus cifras significativas iguales a las de $f(t_M)$. (La diferencia entre este último valor y f(t) depende de la función f que se aproxime: sen, cos, arctan, senh, etc.).

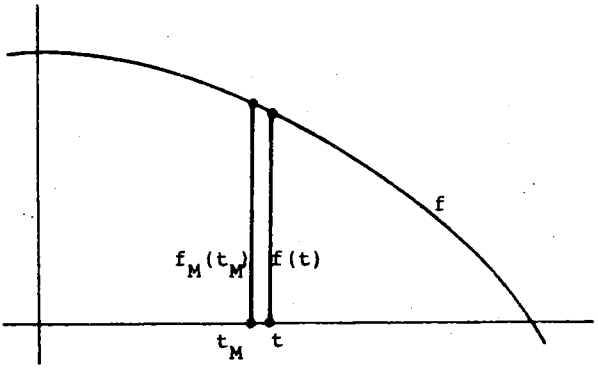


Figura 1. Evaluación de funciones en una máquina. t y f(t) son aproximados por los números de máquina t_M y $f_M(t_M)$, respectivamente.

En la figura 2 se ilustra la forma en que se construye t_M, número que de aquí en adelante denotaremos por t_{n+1} usando para ello una lista decreciente de constantes positivas $\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_n$.

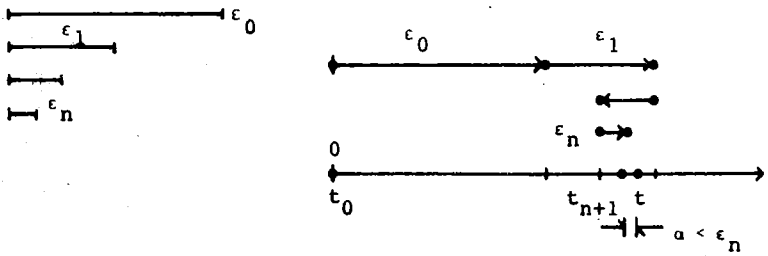


Figura 2. Aproximación de t mediante t_{n+1}, último número de la lista generada por la fórmula $t_{k+1} = t_k + \delta_k \varepsilon_k$, con $t_0 = 0$ y $\delta_k = 1$ si $t_k \leq t$ o no, respectivamente.

Lo interesante es que, escogiendo bien los segmentos, de esta manera es posible acercarse a una distancia menor o igual que ε_n a *cualquier* punto t del intervalo de radio $\varepsilon_0 + \varepsilon_1 + \ldots + \varepsilon_n$ (Figura 3).

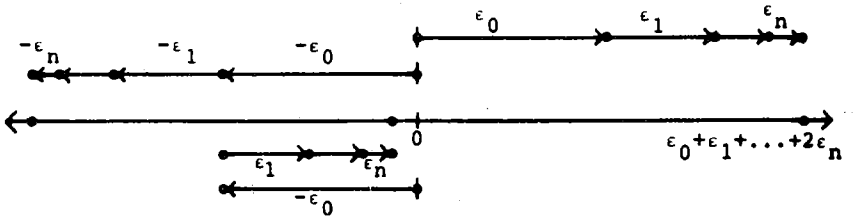


Figura 3. Intervalo de aproximación.

Veamos ahora una forma de escoger los segmentos, es decir, busquemos condiciones suficientes para lograr aproximar t con precisión ε_n.

Para empezar, notemos que no cualquier lista de constantes $\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_n$ permite acercarse a cualquier punto del intervalo. En efecto si, tentados por la idea de acercarnos en pocos pasos, tomamos una lista que decrezca "rápido", nos podemos "quedar cortos"; por ejemplo para un punto como el denotado por t en la Figura 4.

Para evitar este problema la misma figura nos sugiere que, una vez dado el paso ε_0, las constantes restantes deben ser tales que su suma difiera de ε_0 no más de ε_n, es decir, $\varepsilon_0 - \sum_{j=1}^{n} \varepsilon_j \leq \varepsilon_n$; y para no quedar cortos después del paso

ε_1 se debe cumplir que $\varepsilon_1 - \sum_{j=2}^{n}\varepsilon_j \leq \varepsilon_n$; y así sucesivamente para cada una de las constantes ε_k. En resumen, se nos ocurre que cada una de las constantes ε_k. En resumen, se nos ocurre que cada constante ε_k, además de formar parte de una lista decreciente $(\varepsilon_0 \geq \varepsilon_1 \geq \varepsilon_2 \geq \ldots \geq \varepsilon_n > 0)$, debe cumplir la condición siguiente:

(2) $$\varepsilon_k \leq \sum_{j=k+1}^{n} \varepsilon_j + \varepsilon_n.$$

No es difícil demostrar, por inducción sobre k, que esto es suficiente para lograr la aproximación deseada: $|t_{n+1} - t| \leq \varepsilon_n$. En [2] se puede consultar una prueba.

Pasemos a considerar la forma en que se aproximan las funciones elementales; trataremos en detalle el caso de lafunciones coseno y seno. En lo que sigue, pensemos en t como un ángulo en el plano cartesiano, y que uno de sus lados se mueve hacia el otro mediante giros de magnitudes $\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_n$ (Figura 6).

Ahora recordemos las fórmulas que dan las coordenadas $(\bar{x},\bar{y})$ del punto que se obtiene de (x,y) mediante una rotación θ:

$$\bar{x} = x\cos\theta - y\,\mathrm{sen}\theta$$
$$\bar{y} = x\,\mathrm{sen}\theta + y\cos\theta .$$

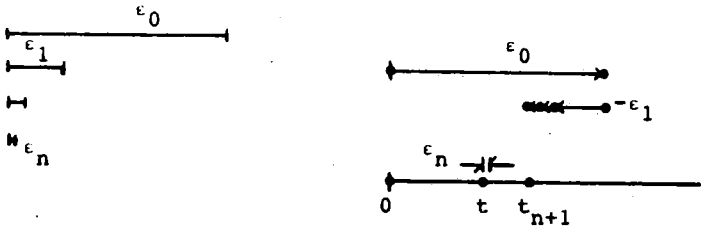


Figura 4. Si la lista $\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_n$ decrece "muy rápido algunos puntos no pueden ser aproximados por t_{n+1} a una distancia menor que ε_n.

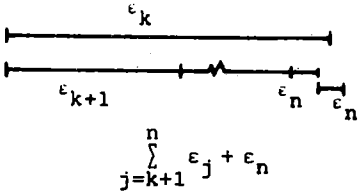


Figura 5

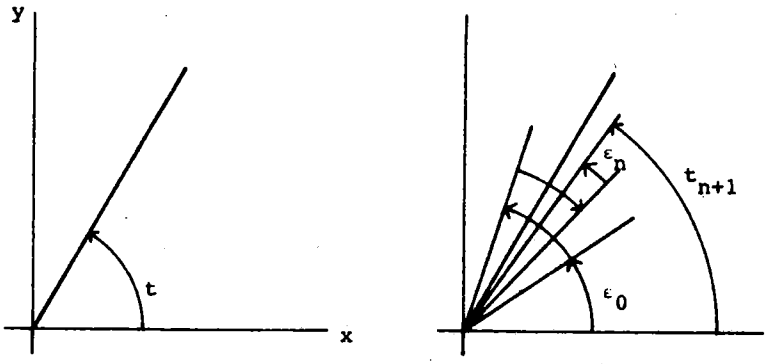


Figura 6. Aproximación del ángulo t mediante t_{n+1}, con $t_0 = 0$ y $t_{k+1} = t_k + \delta_k \varepsilon_k$, $\delta_k = \pm 1$ si $t_k < t$ o no, respectivamente.

Si se aplican estas fórmulas al punto $(\bar{x}_k, \bar{y}_k)$, con $\theta = \delta_k \varepsilon_k$, se obtiene el punto $(\bar{x}_{k+1}, \bar{y}_{k+1})$ (ver Figura 7), en que

$$\bar{x}_{k+1} = \bar{x}_k \cos(\delta_k \varepsilon_k) - \bar{y}_k \operatorname{sen}(\delta_k \varepsilon_k)$$

$$\bar{y}_{k+1} = \bar{x}_k \operatorname{sen}(\delta_k \varepsilon_k) + \bar{y}_k \cos(\delta_k \varepsilon_k)$$

Notando que $\delta_k = \pm 1$ y factorizando $\cos \varepsilon_k$, estas ecuaciones se pueden escribir en la forma

$$\bar{x}_{k+1} = \cos \varepsilon_k (\bar{x}_k - \delta_k \bar{y}_k \tan \varepsilon_k)$$

$$\bar{y}_{k+1} = \cos \varepsilon_k (\bar{y}_k + \delta_k \bar{x}_k \tan \varepsilon_k)$$

Escogiendo cada constante ε_k igual a $\arctan 2^{-k}$ obtenemos:

$$\bar{x}_{k+1} = \cos \varepsilon_k (\bar{x}_k - \delta_k \bar{y}_k 2^{-k})$$

$$\bar{y}_{k+1} = \cos \varepsilon_k (\bar{y}_k + \delta_k \bar{x}_k 2^{-k})$$

Por otro lado, la igualdad para el ángulo,

$$t_{k+1} = t_k + \delta_k \varepsilon_k, \quad \delta_k = \begin{cases} 1 \text{ si } t_k \leq t \\ -1 \text{ si } t_k > t \end{cases}$$

se puede cambiar por la igualdad equivalente

$$z_{k+1} = z_k - \delta_k \varepsilon_k, \quad \delta_k = \begin{cases} 1 \text{ si } z_k \geq 0 \\ -1 \text{ si } z_k < 0 \end{cases}$$

con el valor inicial $z_0 = t$ en lugar del valor inicial $t_0 = 0$. Esta igualdad es preferible porque en las máquinas electrónicas resulta más simple realizar —para determinar δ_k — la comparación de z_k con 0 que la comparación de t_k con t.

Reunamos las últimas igualdades para x, y y z:

$$\begin{aligned} \bar{x}_{k+1} &= \cos \varepsilon_k (\bar{x}_k - \delta_k \bar{y}_k 2^{-k}) \\ \bar{y}_{k+1} &= \cos \varepsilon_k (\bar{y}_k + \delta_k \bar{x}_k 2^{-k}) \qquad \delta_k = \begin{cases} 1 \text{ si } z_k \geq 0 \\ -1 \text{ si } z_k < 0 \end{cases} \\ z_{k+1} &= z_k - \delta_k \varepsilon_k \end{aligned} \tag{3}$$

Estas ecuaciones se parecen a las del algoritmo CORDIC (este nombre es una abreviatura, en inglés, de Computadora Digital para Rotación de Coordenadas), la única diferencia está en el coeficiente $\cos \varepsilon_k$, común a las dos primeras fórmulas. Geométricamente hablando, esto quiere decir que las fórmulas del algoritmo CORDIC, además de aplicar el giro a un punto (x,y), (Figura 8a) lo alejan del origen al no multiplicar por $\cos \varepsilon_k$. De aquí que, si aplicamos reiteradamente las fórmulas con el mismo punto inicial (x_0, y_0), al final obtendremos puntos relacionados así:

$$\bar{x}_{n+1} = Kx_{n+1}$$

$$\bar{y}_{n+1} = Ky_{n+1}$$

en que $K = \prod_{j=1}^{n} \cos \varepsilon_j$. Observemos ahora que si el algoritmo CORDIC se aplica con valores iniciales $x_0 = K$ e $y_0 = 0$, (x_{n+1}, y_{n+1}) será igual al punto que se obtiene de (1,0) mediante la rotación pura; pero entonces la distancia del punto final al origen será igual a 1, por lo que $x_{n+1} = \cos(t_{n+1})$ y $y_{n+1} = \text{sen}(t_{n+1})$, obteniéndose así la aproximación buscada (Figura 8b).

Quedó pendiente la prueba de que las constantes $\varepsilon_k = \arctan 2^{-k}$ constituyen una lista decreciente que satisface la condición (2). De que forman una lista decreciente es fácil convencernos al observar que la función arctan es creciente (Figura 9a). Y para probar que cumplen con (2), vía el teorema del valor medio primero comparamos la pendiente de la recta secante $\overline{AB}$ (Figura 9b) con la pendiente de la recta tangente en el punto A, obteniendo la desigualdad

$$\frac{\varepsilon_j - \varepsilon_{j+1}}{2^{-(j+1)}} < \frac{1}{1 + 2^{2(j+1)}},$$

de donde

$$\varepsilon_j - \varepsilon_{j+1} < \frac{2^{j+1}}{2^{2(j+1)} + 1} \qquad (4.1)$$

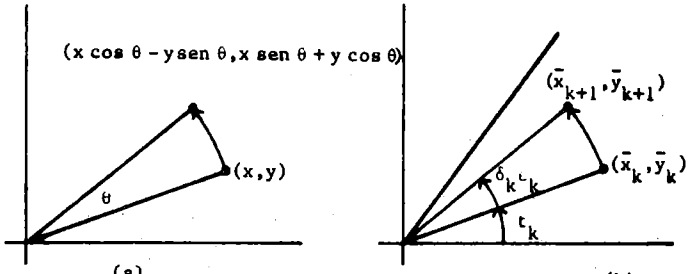


Figura 7. Rotación

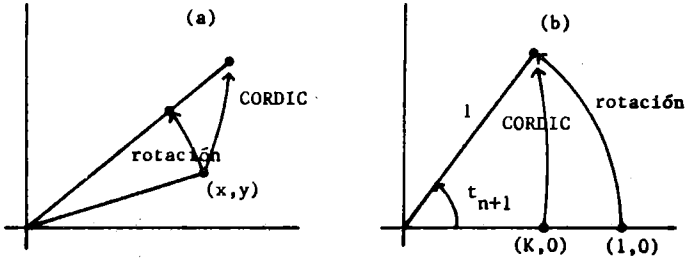


Figura 8. Comparación del algoritmo CORDIC con la rotación.

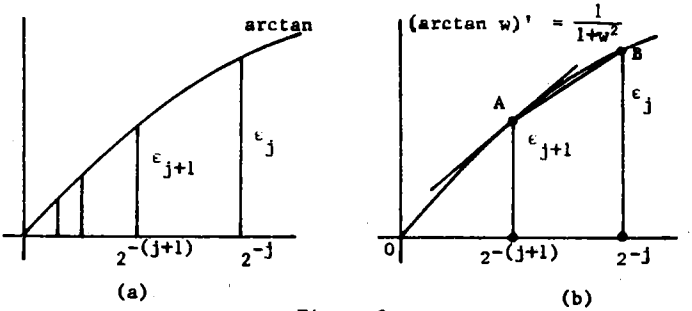


Figura 9

Similarmente, comparando la inclinación de la recta $\overline{OB}$ con la inclinación de la tangente en el punto B, se obtiene la desigualdad

$$\frac{\varepsilon_j}{2^{-j}} < \frac{1}{1+2^{-2j}},$$

equivalente a

$$\varepsilon_j < \frac{2^j}{2^{2j}+1}. \tag{4.2}$$

Aplicando las desigualdades (4.1) y (4.2) justificamos las siguientes:

$$\varepsilon_k - \varepsilon_n = \sum_{j=k}^{n-1} (\varepsilon_j - \varepsilon_{j+1}) < \sum_{j=k}^{n-1} \frac{2^{j+1}}{1+2^{2(j+1)}}$$

$$= \sum_{j=k+1}^{n} \frac{2^j}{1+2^j} < \sum_{j=k+1}^{n} \varepsilon_j$$

de donde se deduce que

$$\varepsilon_k \leq \sum_{j=k+1}^{n} \varepsilon_j + \varepsilon_n$$

En cuanto al tamaño del intervalo de aproximación, es fácil comprobar que $\sum_{j=0}^{3} \varepsilon_j = 1.618 > \Pi/2$, lo que garantiza que los números t de valor absoluto menor que $\Pi/2$ pueden ser aproximados con precisión ε_n, para $n > 3$.

Para aproximar valores de la función arctan mediante (1), se procede en una forma ligeramente distinta a la anterior. En este caso se dan valores iniciales a x e y que correspondan al ángulo que se busca (Figura 10a); z se ini-

cia con $z_0 = 0$, y cada δ_k se escoge con el criterio de que los valores de y se acerquen a 0:

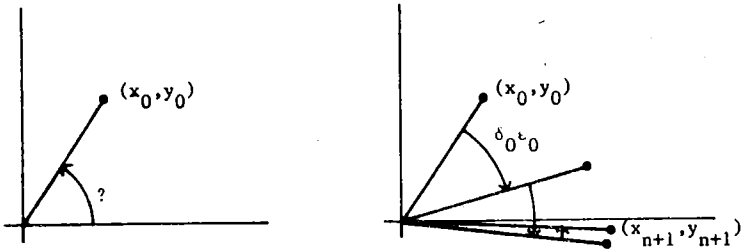
$$\delta_k = \begin{cases} 1 \text{ si } y_k > 0 \\ -1 \text{ si } y_k < 0 \end{cases}$$

De esta manera el rayo que va del origen al punto (x_0,y_0) "barrerá" el ángulo buscado, y la aproximación a éste se obtendrá en z_{n+1} (Figura 10b).

Las fórmulas CORDIC con $m = 1$ permiten aproximar cosh, senh, exp, $\tanh^{-1}$, ln y $\sqrt{\ }$, dando los valores iniciales x_0, y_0, z_0 en forma apropiada y eligiendo cada δ_k en alguna de las dos formas expuestas. Una demostración de que las fórmulas son correctas se basa en las ecuaciones para senh y cosh similares a las aquí dadas para sen y cos. Se presenta un ligero problema con las constantes $\varepsilon_k = \text{arctanh } 2^{-k}$, pues no cumplen la condición (2), pero la dificultad se resuelve repitiéndolas para los índices 4, 13, 40, 121,...,k,3k+1,... [3], determinando así una nueva lista de constantes ε_k, para utilizarse en las fórmulas.

El caso más simple, con $m = 0$ y $\varepsilon_k = 2^{-k}$, permite multiplicar y dividir dos números dados. Así, notando que el valor de x permanece igual a x_0, podemos simplificar (1):

$$y_{k+1} = y_k + \delta_k x_0 2^{-k}$$
$$z_{k+1} = z_k - \delta_k 2^{-k}$$



(a) (b)

Figura 10. (a) Para aproximar arctan w se escogen x_0 e y_0 tales que $y_0/x_0 = w$. (b) Cada δ_k se escoge para que $y_{k+1} = y_k + \delta_k x_k 2^{-k}$ se acerque a 0, al final $z_{n+1} = -\sum_{j=0}^{n} \delta_k \epsilon_k$ aproxima $\arctan(y_0/x_0)$

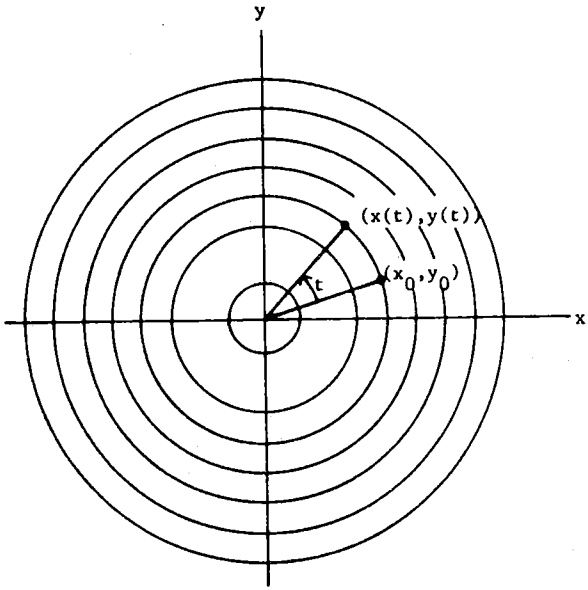


Figura 11. Gráfica de las soluciones en el plano x-y.

De aquí es fácil ver que, con valores iniciales x_0 y z_0, con $y_0 = 0$, y en el modo que fuerza z hacia 0,

$$y_{n+1} = x_0 \sum_{j=0}^{n} \delta_j 2^{-j}$$

$$z_{n+1} = z_0 - \sum_{j=0}^{n} \delta_j 2^{-j} \simeq 0,$$

y entonces $y_{n+1} \simeq x_0 z_0$. Un argumento parecido permite demostrar que, con valores iniciales x_0 e y_0, con $z_0 = 0$ y forzando y hacia 0, se obtiene $z_{n+1} \simeq y_0/x_0$.

En [3] se puede encontrar una demostración elegante —el mismo argumento para los tres casos $m = -1,0,1$— de que el algoritmo CORDIC es correcto.

2. Una Comparación del Algoritmo CORDIC con el Método de Euler

La comapración presentada a continuación fue sugerida por el Dr. Luis Verde Star. Desde el terreno de las ecuaciones diferenciales, el algoritmo CORDIC se puede ver como sigue. Consideremos la ecuación diferencial

$$y'' + my = 0 \tag{5}$$

que podemos plantear equivalentemente en forma de sistema:

$$x' = -my$$
$$y' = x$$

Tratemos el caso en que $m = 1$, con condiciones iniciales

$$x(0) = 1, \quad y(0) = 0$$

Sabemos que $x = \cos t$ y $y = \operatorname{sen} t$ son las soluciones a este problema, y que las fórmulas para aproximarlas mediante el método de Euler, con paso variable h_k, son las siguientes:

$$x_{k+1} = x_k - y_k h_k$$
$$y_{k+1} = y_k + x_k h_k$$
$$t_{k+1} = t_k + h_k$$

con valores iniciales $x_0 = 1$ e $y_0 = 0$.

Como vimos en la sección anterior, el algoritmo CORDIC aproxima estas mismas funciones mediante la aplicación de las fórmulas

$$x_{k+1} = x_k - \delta_k y_k \tan \varepsilon_k$$
$$y_{k+1} = y_k + \delta_k x_k \tan \varepsilon_k$$
$$t_{k+1} = t_k + \delta_k \varepsilon_k$$

con valores iniciales $x_0 = K = \prod_{j=0}^{n} \cos \varepsilon_j$ e $y_0 = 0$ (eligiendo δ_k según se explicó y con $\varepsilon_k = \arctan 2^{-k}$).

Comparando los últimos dos juegos de fórmulas, podemos pensar el algoritmo CORDIC como un método de Euler con paso variable $h_k = \delta_k \varepsilon_k$, y dos ajustes: uno en el valor inicial x_0 y otro en el incremento de las variables x e y, logrado este último al sustituir $\delta_k \varepsilon_k$ por $\tan(\delta_k \varepsilon_k)$ en las dos primeras fórmulas.

Veamos ahora una interpretación geométrica. Recordemos que la solución general de (5) se puede expresar así:

$$x(t) = c_1 \cos t - c_2 \operatorname{sen} t$$
$$y(t) = c_1 \operatorname{sen} t + c_2 \cos t$$

y la solución al problema con valores iniciales $x(0) = x_0$ e $y(0) = y_0$ está dada por

$$x(t) = x_0 \cos t - y_0 \operatorname{sen} t$$
$$y(t) = x_0 \operatorname{sen} t + y_0 \cos t$$

Eliminando el parámetro t, obtenemos la siguiente relación entre $x(t)$ e $y(t)$:

$$x(t)^2 + y(t)^2 = x_0^2 + y_0^2,$$

que graficada en el espacio de fase corresponde a la cir-

cunferencia con centro en el origen y que pasa por el punto (x_0, y_0), como se muestra en la Figura 11.

En forma vectorial, con $W = (x,y)$, la fórmula del método de Euler se puede escribir así:

$$W_{k+1} = W_k + h_k W_k' \tag{6}$$

Para el algoritmo CORDIC la ecuación correspondiente es:

$$W_{k+1} = W_k + \delta_k 2^{-k} W_k' \tag{7}$$

Como W_k' es tangente en el punto W_k a la curva solución, la fórmula anterior nos permite precisar geométricamente la forma en que se genera el punto W_{k+1} a partir del punto W_k: avanzar lo indicado por el vector tangente $\delta_k 2^{-k} W_k'$. En la Figura 12a ilustramos esto para el caso en que se parte del punto $(K,0)$, y que, como vimos antes, genera al punto $W_{k+1} = (x_{k+1}, y_{k+1})$, aproximación de (cos t, sen t). La Figura 12b es la correspondiente a la aproximación de arctan (y_0/x_0).

Procediendo en forma análoga a la anterior, para el caso $m = -1$ las soluciones particulares resultan ser

$$x(t) = x_0 \cosh t + y_0 \operatorname{senh} t$$
$$y(t) = x_0 \operatorname{senh} t + y_0 \cosh t$$

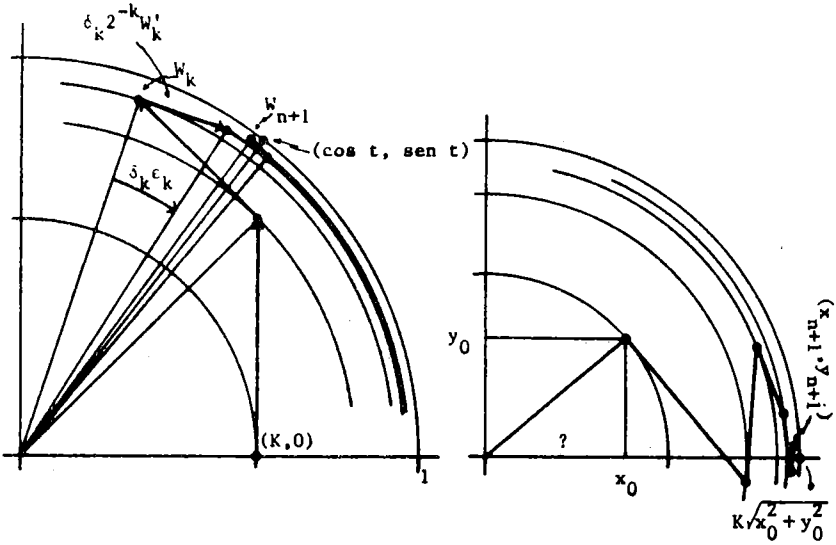


Figura 12

En el espacio fase las curvas solución son las hipérbolas

$$x(t)^2 - y(t)^2 = x_0^2 - y_0^2$$

El parámetro t se puede interpretar geométricamente en la forma siguiente. Usando la fórmula que del teorema de Green se deduce para calcular el área de la región mostrada en la Figura 13,

$$A = \frac{1}{2} \int_c x dy - y dx = \frac{1}{2} \left(x_0^2 - y_0^2 \right) t$$

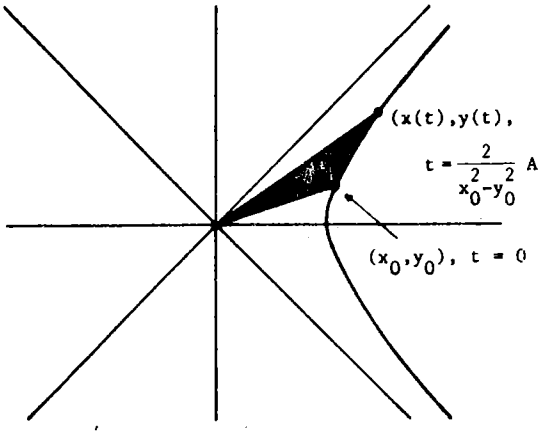


Figura 13

Las fórmulas correspondientes a (6) y (7) resultan ser las mismas, con la aclaración hecha con anterioridad acerca de la doble evaluación para la lista de constantes $\varepsilon_{k'}$.

En la Figura 14a se ilustra la manera de aproximar cosh(t), a partir de los valores iniciales $x_0 = K'$ e $y_0 = 0$, siguiendo el camino marcado por los vectores tangentes a las curvas solución.

3. Conclusiones

Se ha presentado el algoritmo CORDIC para calcular algunas funciones elementales de gran interés en forma unificada. Dicho algoritmo fue ya presentado en [3], donde se

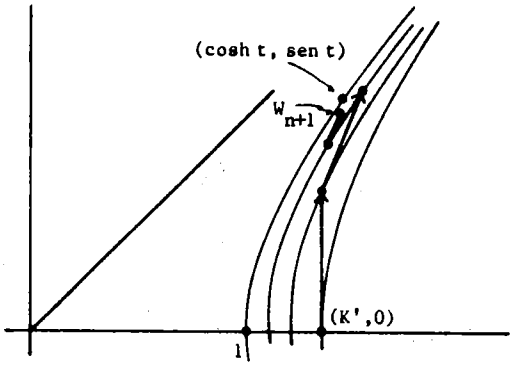


Figura 14

prueba su convergencia utilizando técnicas de variable compleja. Aquí hemos procurado dar una demostración simplificada del mismo hecho, que no requiere sino herramientas elementales. Por otro lado, con ánimo de clarificar las motivaciones subyacentes al algoritmo, hemos incluído una comparación de éste con el que resulta de aplicar el método de Euler a la solución de una ecuación diferencial idonea.

BIBLIOGRAFIA

[1] Haviland, G. and Tuszynski, A; A CORDIC Aritmetic Processor Chip, IEEE Trans. Computers, V. C-29, No. 2 (February 1980), 68-79.

[2] Schelin, Ch.; Calculator Function Approximation, Mathematical Monthly, V. 90, No. 5 (May 1983), 317-325.

[3] Walther, J.; A Unified Algorithm for Elementary Functions, Joint Computer Conference Proceedings, V. 38, Spring 1971, 379-385.

TECNICAS PARA EL MANEJO DE MATRICES RALAS*

Virginia Abrín Batule**

Resumen

Sea $A = (a_{ij})$ una matriz simétrica y definida positivamente. El ancho de banda β de A se define como:

$$\beta = \max |i-j| : a_{ij} \neq 0 .$$

Si la matriz tiene ancho de banda pequeño el problema de resolver un sistema de ecuaciones Ax = b se simplifica considerablemente.

En este trabajo se presentan dos algoritmos: uno para reducir el ancho de banda de una matriz, por medio de intercambios simétricos de renglones y columnas (E. Cuthill y J. McKee) y el otro para reducir el llenado de una matriz (J. Ross), al aplicar eliminación gaussiana.

Introducción

Sea Ax = b un sistema de ecuaciones, en donde A es una matriz simétrica, definida positivamente y rala; e.d. A con-

* Agradezco a Eduardo Rivera y a Diego Bricio Hernández las sugerencias y comentarios al presente trabajo.
** Departamento de Matemáticas, Facultad de Ciencias, UNAM. Departamento de Matemáticas de C.B.I, UAM Iztapalapa.

tiene una gran cantidad de elementos iguales a cero.

Aplicar el método de Cholesky para resolver $Ax = b$ [7], consiste en factorizar A en $A = LL^T$, donde L es una matriz triangular inferior con elementos mayores que cero en la diagonal.

Resolver el sistema $Ax = b$, es equivalente a resolver los sistemas $Ly = b$ y $L^Tx = y$ que son fáciles de resolver, por iteración directa e inversa respectivamente [7]. Es claro que estos sistemas son más sencillos si la parte bajo la diagonal principal de la matriz L contiene un gran número de elementos iguales a cero.

Desafortunadamente, el hecho de que la matriz A sea rala no garantiza que existan muchos elementos iguales a cero en la matriz $F = L + L^T$.

Considérese el ejemplo siguiente.

Sea A_0 como se describe:

$$A_0 = \begin{bmatrix} x & x & x & x & x \\ x & x & & & \\ x & & x & & \\ x & & & x & \\ x & & & & x \end{bmatrix}$$

donde x denota un elemento distinto de cero. Nótese que A_0 es rala. Sin embargo, un cálculo sencillo muestra que el factor L_0 de Cholesky de la matriz A_0 tiene la forma

$$L_0 = \begin{bmatrix} x & & & & \\ x & x & & & \\ x & x & x & & \\ x & \otimes & \otimes & x & \\ x & \otimes & \otimes & \otimes & x \end{bmatrix}$$

donde $\otimes$ denota un elemento que puede ser distinto de cero. El fenómeno observado se conoce como *llenado* (Véase la Sección IV).

De esta forma se ve que la ventaja de tener una gran cantidad de ceros en la matriz A, sólo simplifica el procedimiento para encontrar el factor L de Cholesky y que no necesariamente reduce la complejidad de los sistemas Ly = b y $L^T x = y$. Sin embargo, en ocasiones se puede encontrar un sistema $\bar{A}\bar{x} = \bar{b}$, equivalente al sistema Ax = b, de tal manera que al factorizar $\bar{A}$ como $\bar{A} = \bar{L}\bar{L}^T$, la matriz $\bar{L}$ tenga a lo más tantos elementos distintos de cero como la parte inferior de $\bar{A}$ [2]. Por lo pronto, observemos que para reordenar las ecuaciones, basta premultiplicar ambos miembros del sistema

$$Ax = b$$

por una matriz P de permutaciones, obteniendo

$$PAx = Pb.$$

A su vez, reordenar x con la misma permutación equivalente a reemplazarlo por Px. Dado que

$$PP^T = P^TP = I$$

es posible reescribir el sistema reordenado en la forma de

$$(PAP^T)(Px) = Pb.$$

Así pues, una vez encontrado un ordenamiento P adecuado, se permutan los elementos de b y las hileras de A según P, se reordenan las columnas de A según P^T y se resuelve el sistema resultante, que será

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

en donde

$$\bar{A} = PAP^T, \quad \bar{b} = Pb.$$

La solución x del sistema original, se obtiene entonces permutando las componentes de $\bar{x}$ según P^{-1}.

Sea $A_0x = b_0$ como en el ejemplo anterior y considérese la matriz de permutaciones.

$$P = \begin{bmatrix} . & . & 1 & . & . \\ . & 1 & . & . & . \\ 1 & . & . & . & . \\ . & . & . & 1 & . \\ . & . & . & . & 1 \end{bmatrix}$$

De lo anterior se tiene que resolver el sistema $A_0x = b_0$ es equivalente a resolver $\bar{A}_0\bar{x} = \bar{b}_0$, $\bar{x} = Px$, donde

$$\bar{A}_0 = PA_0P^T = \begin{bmatrix} x & & x & & \\ & x & x & & \\ x & x & x & x & x \\ & & x & x & \\ & & x & & x \end{bmatrix}$$

La factorización de Cholesky de $\bar{A}_0$ tiene como primer factor a

$$\bar{L}_0 = \begin{bmatrix} x & & & & \\ & x & & & \\ x & x & x & & \\ & & x & x & \\ & & x & \otimes & x \end{bmatrix}$$

según se comprueba fácilmente.

Claramente, el número de elementos de L_0 iguales a cero puede ser mayor que el de L_0, por lo que resolver $L_0y = Pb_0$ y $L_0^TPx_0 = y$ posiblemente sea más simple que resolver $L_0y = b$ y $L_0^Tx = y$.

Para describir la propiedad de la matriz A que garantice la raleza de la parte inferior del factor L de A se introducen algunos conceptos.

Definición. El *ancho de banda* de una matriz $A = (a_{ij})$ es $\beta(A) = \max\{|i-j| : a_{ij} \neq 0\}$.

En el ejemplo, A_0 y L_0 tienen ancho de banda 4, mientras que $\bar{A}_0$ y $\bar{L}_0$ tienen ancho de banda igual a dos.

En general se tiene lo siguiente:

Proposición 1. Sean A una matriz simétrica y definida positivamente, y L tal que $A = LL^T$. Entonces $\beta(L) = \beta(A)$.

Demostración. El resultado es claro al aplicar eliminación gaussiana a la matriz A para obtener L^T.

Para una demostración amplia ver A. George [3].

Así pues, si existe una matriz P de permutaciones tal que $\beta(PAP^T) < \beta(A)$ entonces $\beta(\bar{L}) < \beta(L)$, donde $PAP^T = \bar{L}\bar{L}^T$ y $A = LL^T$. Dado que el número de elementos iguales a cero puede ser considerablemente mayor en $\bar{L}$ que en L, se facilita así la solución del sistema $Ax = b$ resolviendo el sistema equivalente $(PAP^T)Px = Pb$.

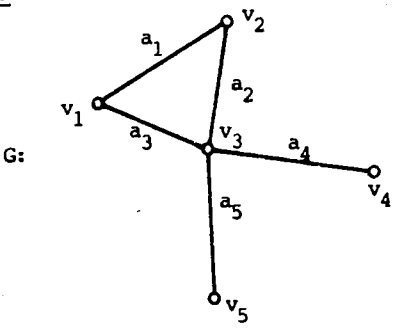
Para encontrar una matriz de permutaciones P adecuada que reduzca el ancho de banda de la matriz A, se dispone de técnicas basadas en la teoría de gráficas que son de gran utilidad [1].

II. Gráficas y Matrices [6].

Una *gráfica* consta de dos conjuntos V y A y una función ϕ. Si se denota por G a una gráfica, V(G) es el conjunto de *vértices* de la gráfica, que es no vacío; A(G) es el conjunto de *aristas* de G y ϕ_G es la función que a cada arista le aso-

cia sus extremos; e.d. $\varphi_G: A(G) \to 2^{V(G)}$, con $\psi_G(a)$ un subconjunto no vacío de V(G) que tiene a lo más dos elementos.

Ejemplo.



$V(G) = \{v_1, v_2, v_3, v_4, v_5\}$

$A(G) = \{a_1, a_2, a_3, a_4, a_5\}$

$\phi(G): A(G) \to 2^{V(G)}$

$\phi(a_1) = \{v_1, v_2\}$, $\phi(a_2) = \{v_2, v_3\}$, $\phi(a_3) = \{v_1, v_3\}$,

$\phi(a_4) = \{v_3, v_4\}$, $\phi(a_5) = \{v_3, v_5\}$.

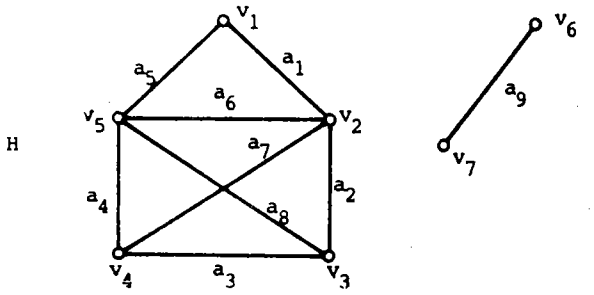
Dos aristas son *incidentes* si tienen un extremo en común. Dos vértices son *adyacentes*, si existe alguna arista que los una.

El *grado* de un vértice v es el número de aristas que lo tienen como extremo. Se denota por gr(v).

Una *trayectoria* en una gráfica es una sucesión alternada de vértices y aristas, de tal manera que ningún vértice se repite y por lo tanto ninguna arista.

Una gráfica G es *conexa* si para todo par de vértices existe una trayectoria que los une.

Ejemplos. Sea H la gráfica:



En la gráfica H, $v_1a_1v_2a_2v_3a_3v_4a_4v_5$ es una trayectoria.

Las aristas a_1 y a_2 son incidentes en v_2, los vértices v_4 y v_2 son adyacentes. Además, $gr(v_2) = gr(v_5) = 4$ y $gr(v_4) = gr(v_3) = 3$.

La gráfica formada por $\{v_1,v_2,v_3,v_4,v_5\}$ y $\{a_1,a_2,a_3,\ldots,a_8\}$ es conexa, mientras que la gráfica formada por $\{v_1,v_2,\ldots,v_5,v_6,v_7\}$ y $\{a_1,a_2,\ldots,a_8,a_9\}$ no lo es.

Las gráficas que se utilizarán en el presente trabajo serán conexas. De hecho, no hay ninguna pérdida de generalidad si la gráfica no es conexa, pues en este último caso los resultados se pueden aplicar en cada una de las componentes de la gráfica.

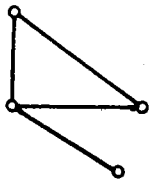
Sea G una gráfica con n vértices, $V(G) = \{v_1,v_2,\ldots,v_n\}$.

Una función $f:V(v) \to \{1,2,\ldots,n\}$ se llama una *numeración* de G, si f es biyectiva.

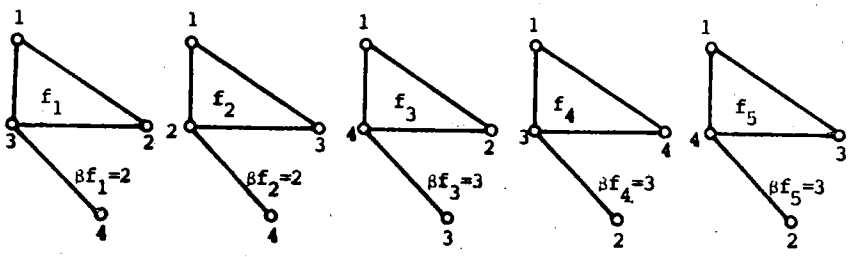
Para cada numeración f de G se define $\beta_f(G)$ como el ancho de banda de G *relativa* a f, en donde

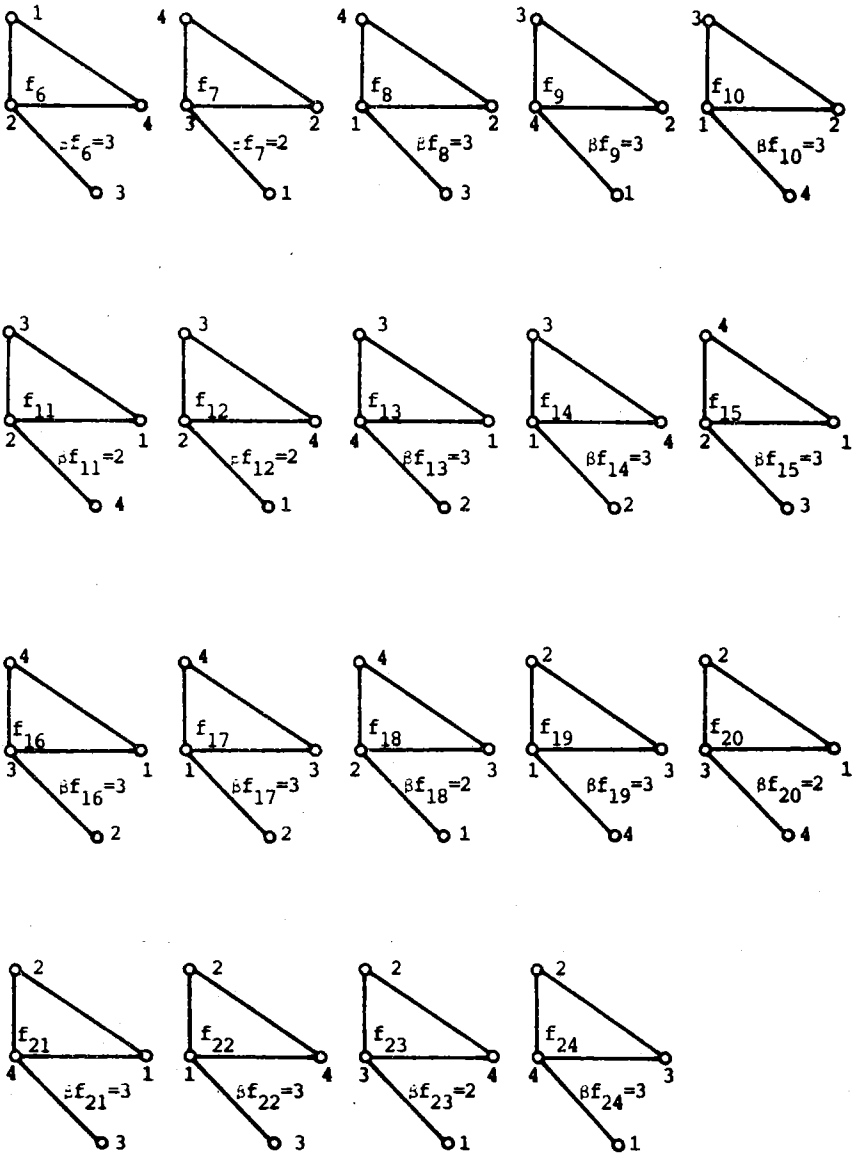
$$\beta_f(G) = \max\{|f(v_i) - f(v_j)| : \{v_i, v_j\} \in \phi_G(A(G))\} .$$

Ejemplo. Sea G la gráfica



Calculando todas las posibles numeraciones de G tenemos en total 24 y están dadas por:




1
f6
2
4
βf6=3
3
4
f7
3
2
βf7=2
1
4
f8
1
2
βf8=3
3
3
f9
4
2
βf9=3
1
3
f10
1
2
βf10=3
4
3
f11
2
1
βf11=2
4
3
f12
2
4
βf12=2
1
3
f13
4
1
βf13=3
2
3
f14
1
4
βf14=3
2
4
f15
2
1
βf15=3
3
4
f16
3
1
βf16=3
2
4
f17
1
3
βf17=3
2
4
f18
2
3
βf18=2
1
2
f19
1
3
βf19=3
4
2
f20
3
1
βf20=2
4
2
f21
4
1
βf21=3
3
2
f22
1
4
βf22=3
3
2
f23
3
4
βf23=2
1
2
f24
4
3
βf24=3
1

Definimos el ancho de banda de G, $\beta(G)$ como:

$$\beta(G) = \min_{f \in N} \beta_f(G)$$

En el ejmplo anterior, $\beta(G) = 2$ y las numeraciones f_1, f_2, f_7, f_{11}, f_{12}, f_{18}, f_{20} y f_{23} son óptimas.

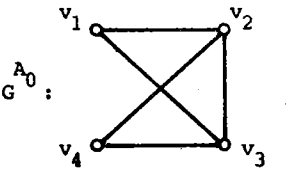
En general, si lo que se desea es reducir el ancho de banda $\beta_f(G)$ de G, se dirá de una numeración f* que es *óptima* si

$$\beta_{f^*}(G) = \beta(G).$$

Pero ¿para qué interesa reducir la banda de una gráfica? Resulta que a cada matriz simétrica $A = (a_{ij})$ de orden n podemos asociarle una gráfica G^A con n vértices $v_1, v_2, \ldots, v_n$, en donde $\{v_i, v_j\}$ es una arista de G si y solo si $a_{ij} \neq 0$.

Ejemplo.

$$A_0 = \begin{bmatrix} x & x & x & . \\ x & x & x & x \\ x & x & x & x \\ . & x & x & x \end{bmatrix}$$



Claramente $\beta(A) = \beta_I(G^A)$ donde I es la numeración de G^A dada por $I(v_i) = i$, con $i = 1, 2, \ldots, n$.

Teorema. Sea A una matriz simétrica y G^A la gráfica correspondiente. Para cada numeración f de G^A, la matriz de permutaciones $P_f = (P_{ij})$ dada por:

$$P_{ij} = \begin{cases} 1 \text{ si } f(v_j) = i \\ \\ 0 \text{ si } f(v_j) \neq i \end{cases}$$

es tal que $\beta(P_f A P_f^T) = \beta_f(G^A)$.

El resultado es claro, pues la manera de definir P_f hace que la permutación de renglones y columnas de A sea la misma permutación de los vértices de G^A definida por la numeración f.

Así pues, si queremos encontrar una matriz P de permutaciones tal que $\beta(PAP^T) < \beta(A)$ basta con encontrar una numeración f de G^A tal que $\beta_f(G^A) < \beta_I(G^A)$. He aquí el origen de nuestro interés por encontrar renumeraciones de los vértices de una gráfica que reduzcan su ancho de banda.

Por cierto, es claro que no es necesario que A sea simétrica para asociarle una gráfica de la manera indicada: basta que su sombra lo sea, donde por *sombra* entendemos la localización de sus elementos distintos de cero. Sin embargo, el caso de A simétrica es de gran interés práctico [4].

En la práctica no es frecuente que pueda lograrse una numeración óptima, pero sí es factible dar renumeraciones subóptimas y efectivas. Una manera de encontrarlas es utili-

zando el algoritmo de E. Cuthill y J. McKee, que no por ser heurístico deja de ser una herramienta poderosa para encontrar renumeraciones buenas y de aquí reducir el ancho de banda. Existen algunos otros algoritmos como los que presentan Norman E. Gibbs, William G. Poole, Jr. y Paul K. Stockemeyer en [5] que, por apoyarse en el algoritmo de E. Cuthill y J. McKee y conceptos de la teoría de las gráficas, son más eficaces para encontrar renumeraciones.

III. Algoritmo de E. Cuthill y J. McKee

En esta sección se dará una descripción de este algoritmo, que junto con un ejemplo ilustrará el funcionamiento de éste. Para esto es importante introducir el concepto de estructura de nivel de una gráfica.

Definición. Una *estructura de nivel* L(G) de una gráfica G, es una partición $V_1, V_2, \ldots, V_n$ de V(G) que satisface:

a) Los vértices de V_1 son adyacentes a los de V_1 y/o a los de V_2.

b) Para $1 < i < n$ los vértices de V_i son adyacentes a los de V_{i-1} y/o V_i y/o V_{i+1}.

c) Si $i = n$, los vértices de V_i son adyacentes a los de V_n y/o V_{n-1}.

A cada vértice $v \in V(G)$ le corresponde una estructura de nivel $L_v(G)$ llamada la *estructura de nivel cimentada en el vértice* v. Los niveles están determinados por:

a) $V_1 = \{v_1\}$

b) Para $i > 1$, V_i es el conjunto de todos los vértices adyacentes a los del nivel V_{i-1} que no han sido colocados en ningún otro nivel.

Para cualquier estructura de nivel L(G), cimentada o no, $W_i(L) = |V_i|$ se llama la *anchura del nivel* i, en donde $|V_i|$ es el número de elementos del conjunto V_i. La *anchura de la estructura* de nivel L(G) se define como $W(L) = \max\{W_i : i = 1,\ldots,n\}$.

Descripción del Algoritmo

Supongamos que la gráfica es conexa, pues de otra forma el algoritmo se puede aplicar a cada una de las componentes de la gráfica.

A) Generar la estructura de nivel en cada vértice v cuyo grado, $gr(v) \leq \text{máx}\{\text{mín}\{(gr_{\text{máx}} + gr_{\text{mín}})/2,\ gr_{\text{promedio}}-1,\ gr_{\text{mín}}\}$. Aquí, grado máximo significa el grado más grande en la gráfica; grado mínimo el menor grado y grado promedio el promedio de los grados de los vértices de la gráfica.

B) Para cada estructura de nivel de anchura mínima generada en A, numerar la gráfica nivel por nivel, con enteros consecutivos y de la siguiente manera:

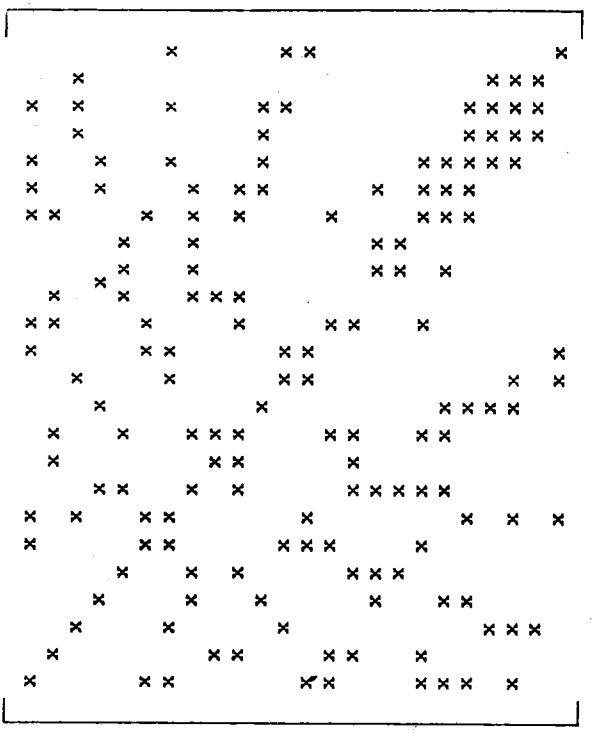
1. Al vértice en donde se cimentó la estructura de nivel asignarle el número 1. Si éste no se encuentra en la primera componente de la gráfica, asignarle el menor

entero positivo que no haya sido asignado.

2. Para los niveles sucesivos empezando por el nivel 2, numerar los vértices adyacentes al vértice 1, en orden creciente de los grados. Los empates se pueden romper de manera arbitraria. Los vértices restantes adyacentes al vértice numerado con el entero positivo menor del nivel anterior, se numeran después en orden creciente de los grados. Se continúa con este procedimiento con todos los vértices del nivel corriente hasta que todos estén numerados; se procede luego a numerar los del nivel siguiente. Cuando todos los vértices han sido numerados, el proceso termina.
3. Para cada numeración f producida en B.2 calcular el ancho de banda $\beta_f(G)$ correspondiente. Seleccionar la numeración que produce el ancho de banda mínimo.

Ejemplo [5]. Sean A una matriz simétrica definida positivamente de 24 x 24 (Figura 3.1) y G la gráfica asociada a A (Figura 3.2).

Los vértices de grado bajo son: el 9, 17, 23 y 24. Por simetría de la gráfica basta considerar la estructura de nivel cimentada en 9 y ésta se genera según el paso A (Figura 3.3). Una nueva numeración se obtiene aplicando el paso B (Figura 3.4) y el cálculo de la banda de la gráfica según la numeración f es $\beta_f(G) = 7$ (paso C); la nueva matriz se mues-



A =

Figura 3.1

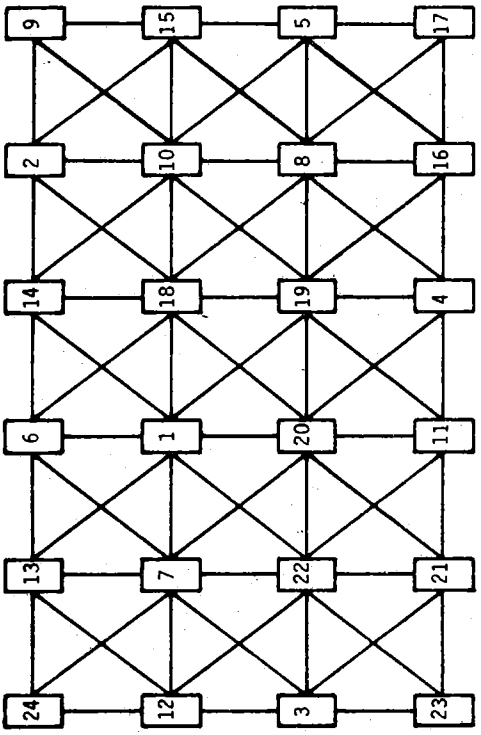


Figura 3.2

La gráfica G asociada a la matriz M (fig. 3.1)

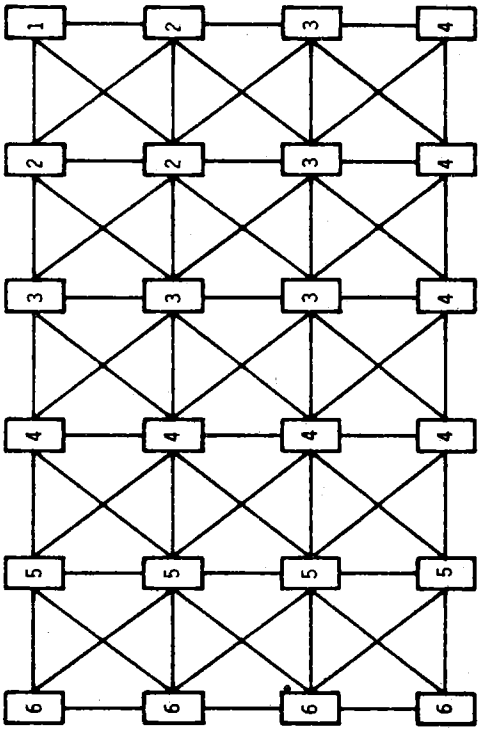


Figura 3.3

La estructura de nivel cimentada en el vértice 9, construida según el paso A.

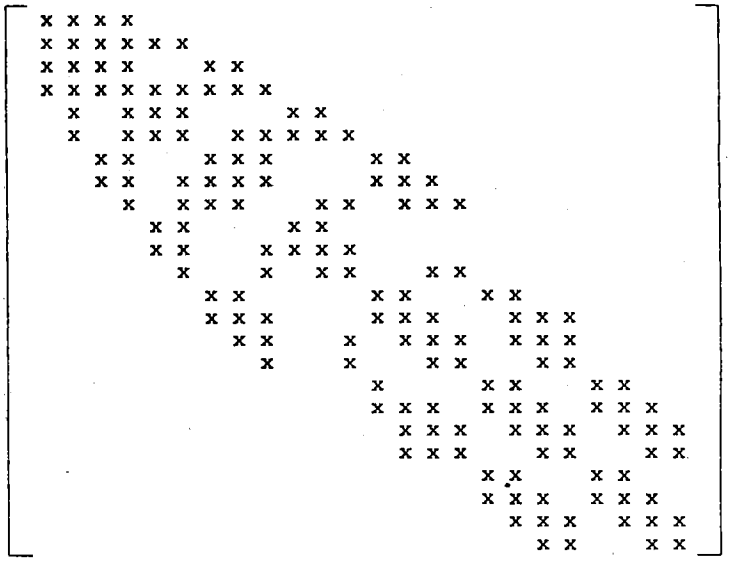


Figura 3.5

La matriz que se obtiene

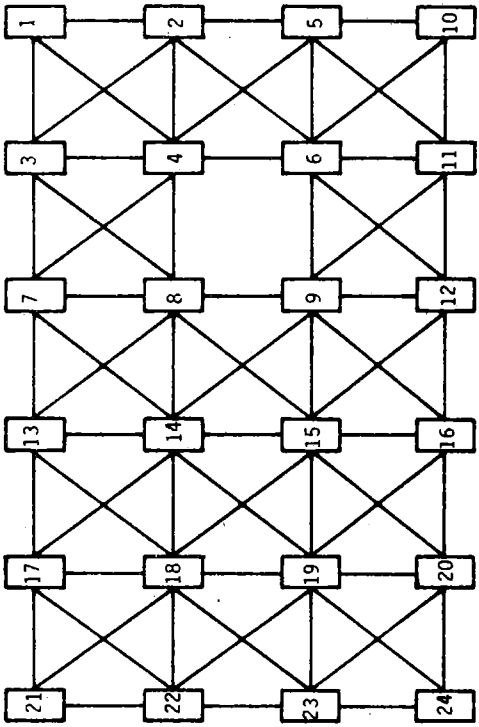


Figura 3.4

La nueva numeración de G obtenida mediante el paso B

tra en la Figura 3.5.

IV. Un algoritmo para reducir el llenado de una matriz

Sea A una matriz simétrica, definida positivamente y rala. Si L es el factor de Cholesky de A y $L+L^T = F$, el *llenado* de A es el número de elementos en F distintos de cero que se localizan en posiciones que en A contenían sólo ceros.

En esta última sección se hará una descripción de la relación que existe entre gráficas, matrices y la eliminación gaussiana. Describiremos también como este proceso nos lleva a construir una sucesión de gráficas $G_0 = G^A, G_1, \ldots, G_n$ a partir de G^A, en donde G_i, se origina en el paso i de la eliminación en la matriz, fijando como pivote a_{ii}. Aquí, $G_i = (V_i, A_i)$ se obtiene quitando a G_{i-1} el vértice v_i, junto con las aristas que lo tienen como extremo y añadiendo las aristas que se forman durante la factorización. Esta sucesión de gráficas y matrices nos lleva a la construcción de la matriz F y la gráfica G^F, en donde se puede ver el llenado de la matriz A y su gráfica correspondiente. Por último, se da un algoritmo [4] llamado *de deficiencia mínima*, el cual permite reducir el llenado de A.

Ejemplo. Consideremos una matriz rala, simétrica y definida positivamente A_0 y la gráfica G_0 correspondiente (Figura 4.1).

La matriz A_1 y la gráfica G_1 se obtienen de A_0 y G_0 res-

pectivamente, tomando a_{11} como pivote para hacer ceros a los elementos de la forma a_{i1} $(i = 2,3,4,5)$ (Figura 4.2). La Figura 4.3 muestra la matriz A_2 y gráfica G_2 que se obtienen de A_1 y G_1 respectivamente, tomando como pivote a_{22}, haciendo ceros los elementos a_{i2} $(i = 3,\ldots,5)$. En el tercer paso consideramos a a_{33} como pivote, se hacen ceros a los elementos de la forma a_{i3} $(i = 4,5)$ y se obtienen la matriz A_3 y la gráfica G_3 (Figura 4.3). El resto del proceso se puede ver en las Figs. 4.4 y 4.5. Una vez obtenidas $A_0, A_1, \ldots, A_4$ y $G_0 G_1 \ldots G_4$ construimos la matriz $F = L+L^T$ y la gráfica G^F; que por ser A simétrica y definida positivamente podemos calcular L, el factor de Cholesky y resulta que la matriz del llenado de A es precisamente F (Figura 4.6).

$$A_0 = \begin{bmatrix} x & x & . & x & . \\ x & x & . & . & x \\ . & . & x & x & x \\ x & . & x & x & x \\ . & x & x & x & x \end{bmatrix}$$

$G^{A_0} = G_0$

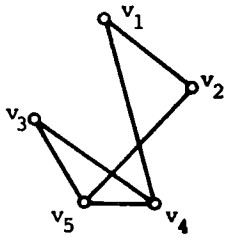


Figura 4.1

$$A_1 = \begin{bmatrix} x & x & . & x & . \\ . & x & . & \otimes & x \\ . & . & x & x & x \\ . & \otimes & x & x & x \\ . & x & x & x & x \end{bmatrix}$$

G_1:

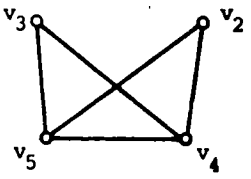


Figura 4.2

⊗ denotan los elementos que pueden ser distintos de cero y que se originan durante la eliminación.

$$A_2 = \begin{bmatrix} x & x & . & x & . \\ . & x & . & \otimes & x \\ . & . & x & x & x \\ . & . & x & x & x \\ . & . & x & x & x \end{bmatrix}$$

G_2:

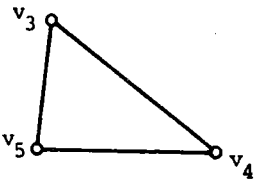


Figura 4.3

$$A_3 = \begin{bmatrix} x & x & . & x & . \\ . & x & . & \otimes & x \\ . & . & x & x & x \\ . & . & . & x & x \\ . & . & . & x & x \end{bmatrix}$$

G_3:

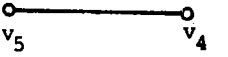


Figura 4.4

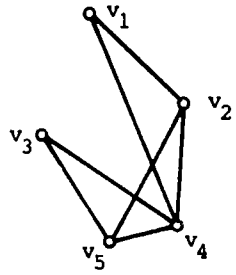
$$A_4 = \begin{bmatrix} x & x & . & x & . \\ . & x & . & \otimes & x \\ . & . & x & x & x \\ . & . & . & x & x \\ . & . & . & . & x \end{bmatrix}$$

G_4:

Figura 4.5

$$F = L+L^T \begin{bmatrix} x & x & . & x & . \\ x & x & . & \otimes & x \\ . & . & x & x & x \\ x & \otimes & x & x & x \\ . & x & x & x & x \end{bmatrix}$$

G^F:



Algoritmo de deficiencia mínima [4]

Definición. Sea G una gráfica y $v \in V(G)$. La *vecindad* N(v) del vértice v es el conjunto de vértices de G adyacentes a v. La *deficiencia* D(v) se define como:

$$D(v) = |\{\{v_j, v_k\} | v_j \in N(v) \text{ y } v_k \in N(v) \text{ y } v_j \notin N(v_k)\}|$$

donde $v_j, v_k \in V(G)$.

En la gráfica de la Figura 4.7 la vecindad de v_4 es $\{v_3, v_6, v_5\}$; la deficiencia de v_1 es 3 y la de v_4 es 2.

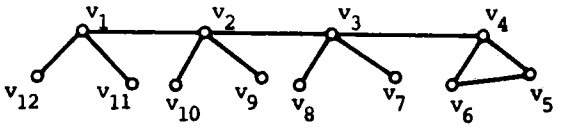


Figura 4.7

El algoritmo de deficiencia mínima es un algoritmo directo, que nos permite encontrar ordenamientos subóptimos para reducir el llenado de una matriz. Una de las ideas principales en el desarrollo de éste, es la de considerar en cada gráfica de eliminación $G_0, G_1, \ldots, G_n$, vértices cuya deficiencia sea mínima; la razón es clara, pues este tipo de vértices introducen menos aristas durante la factorización; de aquí la necesidad de la definición anterior.

Cabe mencionar que si $v_i \in V_{i-1}$ y $D(v_i) = 0$, $G_i = (V_i, A_i)$, se obtiene de $G_{i-1} = (V_{i-1}, A_{i-1})$, eliminando el vértice v_i y las aristas que lo tienen como extremo. Además no se crean nuevas aristas [4].

Descripción del Algoritmo

A) Sea $G_0 = (V_0, A_0)$ una gráfica

1. Elíjase un vértice $v_0 \in V_0$ de tal manera que $|D(v_0)| = \min_{v \in V_0} |D(v)|$ y asígnesele el número 1.
2. Sea G_1 la gráfica que se obtiene a partir de G_0, quitando el vértice v_0, las aristas que son adyacentes a v_0 y añadiendo las nuevas aristas que se forman y considérese un vértice v_1 tal que $|D(v_1)| = \min_{v \in V_1} |D(v)|$ en donde $G_1 = (V_1, A_1)$. Asígnele el número 2 a este vértice.
3. En la i-ésima gráfica $G_{i-1} := (V_{i-1}, A_{i-1})$, que se ob-

tiene como se mencionó al principio de esta sección, elíjase un vértice $v_{i-1} \in V_{i-1}$ que satisfaga: $|D(v_{i-1})| = \min_{v \in V_{i-1}} |D(v)|$ y asígnele el número i-1.

4. Construya la gráfica $G_{i+1} = (V_{i+1}, A_{i+1})$, elíjase un vértice $v_{i+1} \in V_{i+1}$ tal que $|D(v_{i+1})| = \min_{v \in V_{i+1}} |D(v)|$ y asígnesele el número i+1.

5. Si todos los vértices de la gráfica han sido numerados el proceso termina; de otra forma, considérese el conjunto de vértices restantes y elíjase alguno con deficiencia mínima. Constrúyase la gráfica de eliminación correspondiente y asígnesele el menor entero que no haya sido asignado.

Ejemplo. Para ilustrar el algoritmo de deficiencia mínima.

Consideremos la gráfica $G_0 = (V_0, A_0)$ de la Figura 4.7. Los vértices de ésta tienen la misma deficiencia; de aquí que podamos elegir cualquiera de ellos y asignarle el número 1.

Las gráficas de eliminación se muestran en la Figura 4.7, donde los números de la izquierda corresponden a la deficiencia de cada vértice en cada una de las gráficas $G_1, G_2, \ldots, G_6$; y los de la derecha son las etiquetas que tenían en la gráfica original.

La numeración que se asigna a G_0 está dada por el con-

junto de parejas siguiente: $\{(v_1,1),(v_2,4),(v_3,2),(v_4,3),(v_5,5),(v_6,6)\}$.

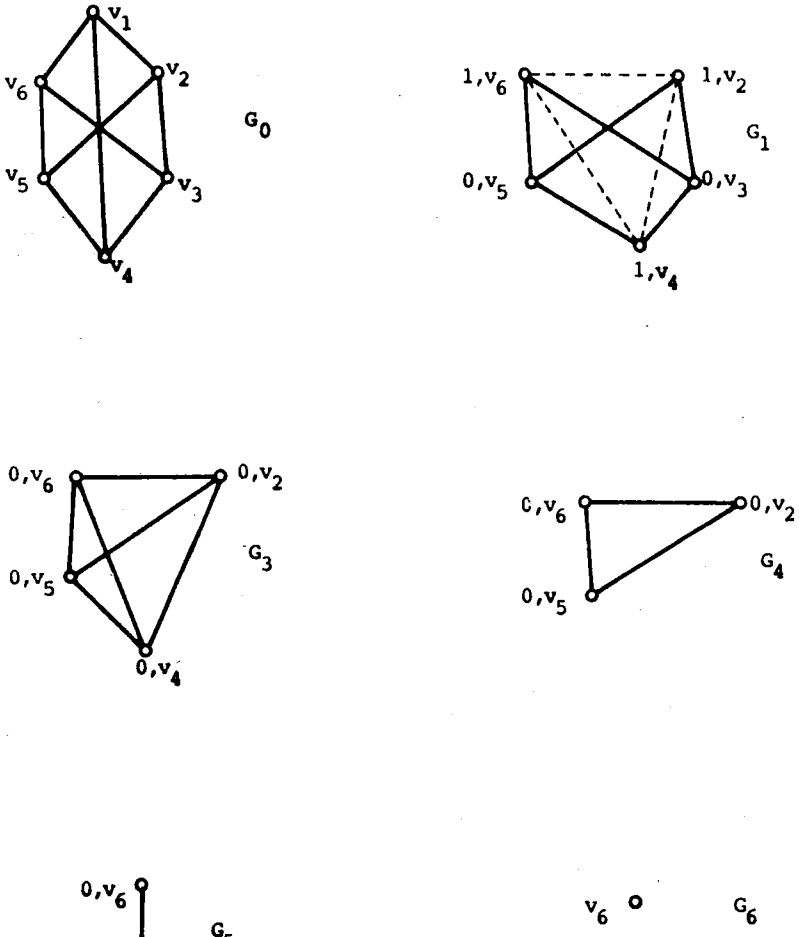


Figura 4.7

Conclusiones

La solución de cierto tipo de problemas en Ingeniería (Cálculo de estructuras, por ejemplo), Economía (matrices de insumo producto), programación lineal y redes físicas lineales [1], etc., en algunos casos queda reducida a resolver un sistema del tipo Ax = b; en donde A es rala, simétrica y definida positivamente. Los algoritmos definidos en las Secciones III y IV son de gran ayuda para resolver sistemas de ecuaciones con las características antes mencionadas. Es necesario hacer hincapié en que este tipo de algoritmos generan numeraciones subóptimas, que no por esta razón dejan de ser útiles para resolver de una manera eficiente los sistemas de ecuaciones con matrices de esta clase.

BIBLIOGRAFIA

1. A. Brameller, R.N. Allan, Y.M. Hamam; Sparsity, Pitman Publishing 1976.
2. J. A. George; Direct Solutions of Sparse Positive Definite Systems: Some Basic Ideas and Open Problems, pp. 283-306. Sparse Matrices and their Uses, Iain S. Duff (ed), Academic Press, Nueva York, 1981.
3. J.A. George, I.W.H. Liu; Computer Solutions of Large Sparse Positive Definite Systems, Prentice Hall, Englewood Cliffs, New Jersey 1981.
4. J.A. George; Computer Implementation of the Finite Element Method, Ph.D. Thesis, University Microfilms International Ann Arbor, Michigan y Londres 1971.
5. N.E. Gibbs, N.G. Poole , Jr., P.K. Stockmeyer; An Algorithm for Reducing the Bandwidth and the Profile of Sparse Matrix, SIAM J. Numer. Anal., 13(1976), pp. 236-250.
6. F. Harary; Graph Theory, Addison Wesley, Reading M.A., 1971, Segunda Edición 1971.
7. Hernández, D.B.; Análisis Numérico: discretización y algorítmica, Notas del Tercer Coloquio del Departamento de Matemáticas del CIEA, IPN, (La Trinidad, Tlax., México) Cd. de México, 1983.
8. Tewarson, R.P.; Sparse Matrices, Academic Press, Nueva York, 1973.

APROXIMACION E INTERPOLACION

Luis Verde Star*

Resumen

Presentamos algunas ideas acerca del problema de aproximación de números, vectores y funciones.

Consideramos la aproximación como substitución de un algoritmo complicado por otro más simple.

Mostramos diversas maneras de medir la distancia entre vectores y funciones que dan lugar a diversos problemas de aproximación.

Vemos también que a través de la discretización que consiste en reemplazar funciones por vectores, aparecen los problemas de interpolación.

Demostramos el teorema básico de la interpolación polinomial, que nos dice cómo encontrar el polinomio de grado mínimo cuya gráfica pasa por un conjunto dado de puntos en el plano.

Concluimos con algunas observaciones acerca del error en la aproximación de funciones derivables por medio de interpolación polinomial.

* Departamento de Matemáticas, Universidad Autónoma Metropolitana, Unidad Iztapalapa.

Existen diversas formas de especificar un número real. Por ejemplo, $\sqrt{2}$ es la raíz positiva de la ecuación $x^2-2=0$, π es el área de un círculo de radio igual a uno, $e = \sum_{n\geq 0} \frac{1}{n!}$, $2/3 = 0.666\ldots$, etc.

Para efectuar operaciones aritméticas con números reales generalmente utilizamos la representación decimal. Si $x \in [0,1]$ entonces $x = 0.\, a_1a_2a_3,\ldots,$ donde cada a_j es un dígito entre 0 y 9, ésto es, $x = \sum_{j\geq 1} a_j 10^{-j}$. Por lo tanto conocer x equivale a conocer la sucesión infinita $a_1, a_2, a_3,\ldots,$ lo cual en ciertos casos es fácil, por ejemplo si $x = 2/3$, pero en otros casos es difícil o muy laborioso, por ejemplo si $x = \pi/4$ ó $x = \sqrt{2}/2$ o si x es la menor raíz de $e^x-3x = 0$. Esto muestra la necesidad de aproximar, ésto es, reemplazar x por un número z que esté cercano a x y que tenga una representación decimal simple.

A cada $x \in [0,1]$ corresponde un algoritmo que genera los dígitos a_j de la representación decimal de x. Si x es racional el algoritmo es relativamente simple porque la sucesión $\{a_j\}$ es finalmente periódica.

Para aproximar $x \in [0,1]$ truncamos su representación decimal. Escogemos un entero positivo m y reemplazamos la sucesión $a_1, a_2, a_3,\ldots$ por la sucesión $a_1, a_2, a_3,\ldots, a_m, 0, 0, 0,\ldots$ Esto equivale a interrumpir la ejecución del algoritmo de x después de m pasos y tomar $a_j = 0$ si $j > m$. En esta forma cada $x \in [0,1]$ es aproximado por un elemento del conjunto

$$A_m = \{r: \exists k \in \mathbb{N} \text{ tal que } k < 10^m \text{ y } r = k/10^m\}$$

y por lo tanto cada real es aproximado por un elemento de $\mathbb{Z} + A_m$.

Notemos que para cada real x existe por lo menos un $s = n + k/10^m$ en $\mathbb{Z} + A_m$ tal que $|x-s| \leq |x-r|$ para todo r en $\mathbb{Z} + A_m$, o sea que s minimiza la distancia entre x y $\mathbb{Z} + A_m$. Además, dado $x \in \mathbb{R}$ y dado $\varepsilon > 0$ existe m y existe $s \in \mathbb{Z} + A_m$ tal que $|x-s| < \varepsilon$. Es claro que la calidad de la aproximación aumenta con m pero también aumenta la complejidad del racional s.

La aproximación de reales por racionales ilustra varias ideas que aparecen en otros procesos de aproximación. Cuando consideramos la aproximación de vectores en $\mathbb{R}^n$ aparecen algunos aspectos geométricos importantes de la teoría de aproximación.

Consideremos el siguiente problema: dado un vector u en $\mathbb{R}^3$ y un plano P que no contiene a u, encontrar un vector v en P tal que la distancia euclidiana entre u y v sea menor o igual que la distancia entre u y cualquier otro elemento de P. La geometría nos muestra que existe siempre una única solución v del problema y que el vector $u-v$ es perpendicular a P.

Si en lugar de un plano consideramos un conjunto cualquiera A que no contenga a u, es posible que no exista solución o que existan varias soluciones. Por ejemplo:

a) $u = (1,1,1)$, $A = \{x \in \mathbb{R}^3 : |x| < 1\}$,

b) $u = 0$, $A = \{x \in \mathbb{R}^3 : |x| = 1\}$.

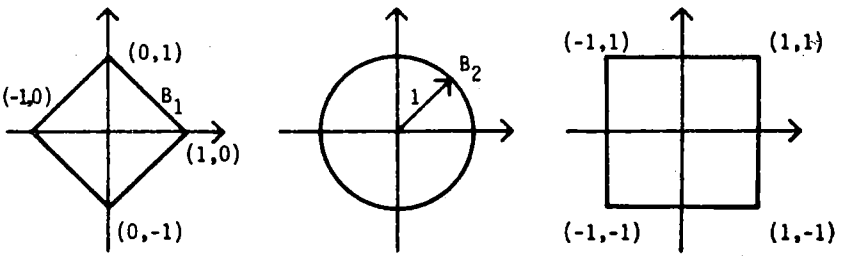
Podemos también substituir la distancia euclidiana en $\mathbb{R}^n$ por alguna otra. Las distancias más usuales son:

$$d_1(x,y) = |x-y|_1 = \sum_{i=1}^{n} |x_i - y_i|$$

$$d_2(x,y) = |x-y|_2 = \left(\sum_{i=1}^{n} (x_i - y_i)^2 \right)^{1/2} \quad \text{(distancia euclidiana)}$$

$$d_\infty(x,y) = \|x-y\|_\infty = \max_{1 \le i \le n} |x_i - y_i|$$

El conjunto $B_i = \{x \in \mathbb{R}^2 : d_i(x,0) \le 1\}$, $i = 1,2,\infty$, se muestra en la figura siguiente



Es claro que dados $A \subseteq \mathbb{R}^n$ y $u \notin A$, el conjunto de soluciones del problema de aproximar u por elementos de A depende del concepto de distancia que se utilice.

Las distancias o métricas en $\mathbb{R}^n$ dan lugar a conceptos de distancia entre funciones. En el espacio $C(I)$ de funciones continuas en el intervalo acotado cerrado I la distancia entre dos funciones f,g puede medirse en varias formas, por ejemplo:

$$\text{i)} \quad |f-g|_\infty = \max_{t\in I} |f(t)-g(t)|$$

$$\text{ii)} \quad |f-g|_2 = \left(\int_I (f(t)-y(t))^2 dt \right)^{1/2}$$

$$\text{iii)} \quad |f-g|_1 = \int_I |f(t)-y(t)|dt.$$

Notemos que estos conceptos de distancia se obtienen de las distancias en $\mathbb{R}^n$ considerando a las funciones $f:I \to \mathbb{R}$ como "vectores" con un número infinito de componentes, una para cada $t \in I$. Es importante notar que la distancia entre f y g puede ser difícil de calcular si $f-g$ no es muy simple.

Consideremos ahora la aproximación de funciones reales.

Sea I un intervalo acotado y cerrado y sea F una clase de funciones definidas en I y con valores reales. Supongamos que d es una métrica en F y que $\{A_\alpha\}$ es una familia de subconjuntos de F. Decimos que la familia $\{A_\alpha\}$ aproxima a F si dados $f \in F$ y $\varepsilon > 0$ existe α y existe

$p \in A_\alpha$ tal que $d(f,p) < \varepsilon$.

Un ejemplo importante se obtiene tomando $F = C(I)$, A_α igual al conjunto de polinomios de grado menor o igual que α y la métrica d_∞ en $C(I)$.

Una función $f:I \to \mathbb{R}$ puede considerarse como un algoritmo que para cada $t \in I$ produce el número real $f(t)$. En algunos casos el algoritmo puede ser complicado o muy laborioso, por ejemplo cuando f está definida por una serie, por una integral definida o por una ecuación diferencial. En estos casos es conveniente aproximar f usando una función p a la que corresponda un algoritmo simple, por ejemplo un polinomio o una función polinomial por secciones, tal que $d(f,p)$ sea pequeña para alguna métrica d en el espacio de funciones.

Debido a que el cálculo de $d(f,p)$ requiere evaluar $f(t)-p(t)$ para cada $t \in I$, en la práctica se toma un conjunto finito $\{t_0,t_1,\ldots,t_n\}$ en I y se busca una función p tal que la distancia entre los vectores $(f(t_0),f(t_1),\ldots,f(t_n))$ y $(p(t_0),p(t_1),\ldots,p(t_n))$ sea pequeña. En ciertos casos esto garantiza que la distancia entre f y p es también pequeña

Reemplazamos la función de error $f-p$ por el vector $e = (e_0,e_1,e_2,\ldots,e_n)$ donde $e_i = f(t_i)-p(t_i)$, y buscamos p tal que $e \equiv 0$ o tal que $d_j(e,0)$ sea mínima para alguna $j = 1,2,\infty$. En el primer caso tenemos un problema de interpolación. Es claro que en general la condición $e \equiv 0$ no implica que $f-p \equiv 0$, pero bajo ciertas hipótesis en la clase de

funciones F, el número n y la distribución de las t_j en I, si $e = 0$ entonces $d(f,p)$ es pequeña y por tanto $|f(t)-p(t)|$ es pequeño para cada $t \in I$.

Si queremos encontrar p que minimice $d_2(f,p)$ obtenemos un problema de mínimos cuadrados.

La parte algebraica del problema de interpolación es la siguiente. Dado $\{t_0,t_1,t_2,\ldots,t_n\}$ subconjunto de I y dado un vector $y = (y_0,y_1,y_2,\ldots,y_n)$ en $\mathbb{R}^{n+1}$ encontrar un polinomio P de grado mínimo tal que $P(t_i) = y_i$, $0 \le i \le n$. Mostraremos que siempre existe una solución P de grado menor o igual que n.

Supongamos que $P(t) = a_0 + a_1 t + \ldots + a_n t^n$. Entonces queremos encontrar los coeficientes a_j tales que

$$a_0+a_1t_i + a_2t_i^2 + \ldots + a_nt_i^n = y_i, \quad 0 \le i \le n. \qquad (*)$$

Este es un sistema de $n+1$ ecuaciones lineales en las incógnitas $a_0,a_1,a_2,\ldots,a_n$. La matriz de coeficientes del sistema es

$$V = \begin{pmatrix} 1 & t_0 & t_0^2 & \cdots & t_0^n \\ 1 & t_1 & t_1^2 & \cdots & t_1^n \\ \vdots & & & & \\ 1 & t_n & t_n^2 & \cdots & t_n^n \end{pmatrix}$$

El sistema (*) puede escribirse en la forma $Va = y$ donde $a = (a_0, a_1, a_2, \ldots, a_n)$.

Usando inducción en n y algunas propiedades de los determinantes podemos demostrar que $\det V \neq 0$ y por lo tanto el sistema (*) tiene una única solución a para cada vector y.

Si utilizamos una idea de Newton podemos simplificar la demostración descrita arriba. La idea consiste en escribir el polinomio P en una forma ligeramente diferente de la usual, con el propósito de obtener un sistema de ecuaciones equivalente pero más simple que (*).

Suponemos que

$$P(t) = b_0 + b_1(t-t_0) + b_2(t-t_0)(t-t_1) + \ldots + b_n(t-t_0)\ldots(t-t_{n-1}).$$

Entonces las condiciones $P(t_i) = y_i$, $0 \leq i \leq n$ nos dan el sistema de ecuaciones

$$b_0 + b_1(t_i-t_0) + \ldots + b_i(t_i-t_0)\ldots(t_i-t_{i-1}) = y_i, \qquad 0 \leq i \leq n. \qquad (**)$$

Notemos que el sistema (**) es triangular, esto es, en la i-ésima ecuación solamente aparecen las incógnitas $b_0, b_1, b_2, \ldots, b_i$. Además el coeficiente de b_i en la i-ésima ecuación es diferente de cero porque las t_j son distintas. Por lo tanto (**) se resuelve a partir de la primera ecuación ($b_0 = y_0$) substituyendo los valores de $b_0, b_1, \ldots, b_{i-1}$

en la i-ésima ecuación y despejando b_i, $i \leq i \leq n$.

Describiremos ahora el método de Lagrange para resolver el problema de interpolación.

Consideramos primero el caso simple $y = (1,0,0,0,\ldots,0)$. Queremos un polinomio que tenga $t_1, t_2, \ldots, t_n$ como raíces y que tome el valor 1 en t_0. El producto $(t-t_1)(t-t_2)\ldots(t-t_n)$ se anula en $t_1, t_2, \ldots, t_n$ y si lo dividimos entre el valor que toma en t_0 obtenemos el polinomio

$$\ell_0(t) = \frac{(t-t_1)(t-t_2)\ldots(t-t_n)}{(t_0-t_1)(t_0-t_2)\ldots(t_0-t_n)}$$

que resuelve el problema para $y = (1,0,0,\ldots,0)$. En la misma forma construimos los polinomios $\ell_j(t)$ tales que

$$\ell_j(t_i) = \begin{cases} 1 & \text{si } j = i \\ 0 & \text{si } j \neq i \end{cases} \qquad 0 \leq i,j \leq n.$$

El principio de superposición nos da la solución al caso general. Dado $y = (y_0, y_1, y_2, \ldots, y_n)$ tomamos el polinomio

$$P(t) = \sum_{j=0}^{n} y_j \ell_j(t)$$

Es claro que $P(t_i) = y_i$, $0 \leq i \leq n$. Como cada ℓ_j es un polinomio de grado n el polinomio P es de grado menor o

igual que n.

Los polinomios ℓ_j son llamados polinomios básicos de Lagrange y constituyen una base del espacio vectorial de polinomios de grado menor o igual que n.

Como el vector y es arbitrario, dada cualquier función $f:I \to \mathbb{R}$ existe un único polinomio $P(t)$ de grado $\leq n$ tal que $P(t_i) = f(t_i)$, $0 \leq i \leq n$. Si no tenemos hipótesis adicionales no podemos decir nada acerca del error $|f(t)-P(t)|$ cuando t no es uno de los puntos de interpolación $t_0, t_1, \ldots, t_n$.

Usando el teorema de Rolle en forma iterada podemos demostrar el teorema siguiente.

TEOREMA. Si f y sus derivadas $f', f'', \ldots, f^{(n+1)}$ son continuas en I, si $\{t_0, t_1, \ldots, t_n\} \subseteq I$ y P es el polinomio que interpola a f en los t_i, entonces para cada $t \in I$ existe un número real z en el intervalo $[\min\{t, t_0, t_1, \ldots, t_n\}, \max\{t, t_0, t_1, \ldots, t_n\}]$ tal que

$$f(t) - P(t) = (t-t_0)(t-t_1)\ldots(t-t_n)\,\frac{f^{(n+1)}(z)}{(n+1)!}$$

y por lo tanto, para todo $t \in I$

$$|f(t) - P(t)| \leq \frac{1}{(n+1)!}\,|t-t_0||t-t_1|\ldots|t-t_n|\,\|f^{(n+1)}\|_\infty .$$

Si $\|f^{(n+1)}\|_\infty = \max_{x \in I} |f^{(n+1)}(x)|$ es pequeño y las distancias $|t-t_i|$ no son grandes, entonces el error $|f(t)-P(t)|$ es pequeño.

Si $\|f^{(n+1)}\|_\infty$ es pequeño entonces en cierta forma f no está lejos del espacio A_n de polinomios de grado $\leq n$, porque si $p \in A_n$ entonces $p^{(n+1)} \equiv 0$. Si $n = 1$, $\|f''\|_\infty$ pequeña significa que la gráfica de f no difiere mucho de una recta, esto es, de la gráfica de un polinomio de grado uno.

Cuando el grado del polinomio de interpolación es muy grande la evaluación del polinomio en un punto requiere muchas operaciones aritméticas y los errores de truncación pueden ser grandes. Una alternativa consiste en dividir el intervalo I en varios subintervalos I_j y encontrar un polinomio p_j de grado pequeño que interpole a la función dada en I_j. En esta forma obtenemos un algoritmo de evaluación muy simple pero que requiere almacenar una mayor cantidad de datos.

Es posible construir una función de interpolación que en cada subintervalo coincida con un polinomio cúbico y que sea derivable dos veces en todo el intervalo I. Esto se hace resolviendo un sistema de ecuaciones lineales que además de las condiciones de interpolación incluye condiciones para que las derivadas de los polinomios cúbicos coincidan en la frontera de los subintervalos.

BIBLIOGRAFIA

1. Peter Henrici; Elementos de Análisis Numérico, Ed. Trillas, México, 1972.

2. J.P. Davis; Interpolation and Approximation, Dover, N.Y., 1975.

RESOLUCION NUMERICA DE LA ECUACION DE BURGERS

Patricia Saavedra B.*

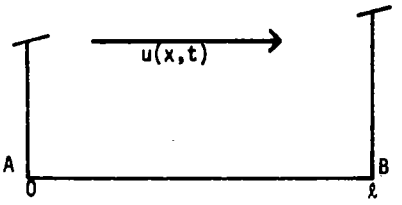
Resumen

El objetivo de esta exposición es aplicar algunos métodos numéricos de residuos pesados o ponderados a la resolución numérica de una ecuación diferencial parcial no lineal que depende del tiempo: la ecuación de Burgers $u_t + uu_x = \nu u_{xx}$. Esta es en general parabólica de segundo orden, pero se reduce a elíptica si nos interesa el caso estacionario $(u_t = 0)$ y a hiperbólica de primer orden si $\nu = 0$. Este último caso presenta dificultades considerables al permitir que se propaguen ondas de choque; para evitar tal caso consideraremos que ν es suficientemente grande como para que el término νu_{xx} sea significativo.

I. Presentación de la ecuación. Formulación variacional.

La ecuación de Burgers se puede aplicar para estudiar el comportamiento del viento dentro de una región limitada por dos estaciones metereológicas A y B.

*Depto. de Matemáticas, UAM, Unidad Iztapalapa.



El comportamiento se describe en general por la ecuación de Navier Stokes [1] que para este caso particular se reduce a la ecuación de Burgers ya que se supone que el flujo es unidimensional, sin caida de presión, etc. Así pues, se trata del siguiente problema de valores iniciales y a la frontera:

$$(P)\quad \begin{cases} \dfrac{\partial u}{\partial t}(x,t) + u(x,t)\dfrac{\partial u}{\partial x}(x,t) = \nu \dfrac{\partial^2 u}{\partial x^2}(x,t) \quad \text{en} \quad <0,\ell>\times<0,t_0> \\ \\ u(0,t) = a(t) \qquad u(\ell,t) = b(t) \\ \\ u(x,0) = r_0(x). \end{cases}$$

Aquí $\nu > 0$ es una constante física (la viscosidad cinemática) función de la temperatura, la presión y otras variables termodinámicas.

Supongamos que $r_0 \in C^0(<0,\ell>)$ y que $b(t)$, $a(t)$ son $C^1(<0,t_0>)$.

Si el problema P admite una solución, ésta deberá estar en el siguiente espacio de funciones:

$$V = \{v:|0,\ell| \times |0,t_0| \to \mathbb{R} \mid v,\ \frac{\partial v}{\partial x},\ \frac{\partial^2 v}{\partial x^2},\ \frac{\partial v}{\partial t}$$

son continuas en $|0,\ell| \times |0,t_0|\}$.

Definamos las siguientes funciones:

$$g:V \to C^0(|0,\ell| \times |0,t_0|)$$

$$g(v) = v\,\frac{\partial v}{\partial x}$$

$$L:V \to C^0(|0,\ell| \times |0,t_0|) \qquad Lv = -\nu\,\frac{\partial^2 u}{\partial x^2}\,(x,t)$$

El problema P puede escribirse de la forma:

$$(P)\ \begin{cases} \dfrac{\partial u}{\partial t} + g(u) + Lu = 0 \quad \text{en} \quad (0,\ell) \times (0,t_0) \\ u(0,t) = a(t) \quad u(\ell,t) = b(t) \\ u(x,0) = r_0(x). \end{cases}$$

Supongamos que a y b son constantes y que $\frac{\partial u}{\partial t} = 0$; entonces, el problema P se reduce a resolver la siguiente ecuación diferencial ordinaria:

$$(P_E)\ \begin{cases} Lu + g(u) = 0 \quad \text{en} \quad (0,\ell) = \Omega \\ u(0) = a \quad u(\ell) = b \end{cases}$$

Si u es solución de P_E, entonces u debe estar en el siguiente espacio de funciones:

$$W = \{u:|0,\ell| \to \mathbb{R} \mid u \in C^2(\Omega)\}.$$

Sea $u_0 \in W$ tal que $u_0(0) = a$ y $u_0(\ell) = b$, entonces si encontramos una $w \in W$ que satisfaga:

$$P_{EH} \begin{cases} L(w+u_0) + g(w+u_0) = 0 \quad \text{en } \Omega \\ \\ w(0) = w(\ell) = 0 \end{cases}$$

Obtendremos que $u = w+u_0 \in W$ y esta función es solución del problema P_E. Por lo que, de aquí en adelante, estudiaremos el problema P_{EH}.

Introduzcamos los siguientes espacios de funciones:

$$D(\Omega) = \{v:\Omega \to \mathbb{R} \mid v \in C^\infty(\Omega) \quad \text{con soporte compacto en } \Omega\}$$

$$L^2(\Omega) = \left\{ v:\Omega \to \mathbb{R} \mid \int_0^\ell |v(x)|^2 dx < \infty \right\}$$

$$H^1(\Omega) = \left\{ v \in L^2(\Omega) \mid \frac{dv}{dx} \in L^2(\Omega) \right\}$$

$$H_0^1(\Omega) = \{v \in H^1(\Omega) \mid v(0) = v(\ell) = 0\}$$

Si definimos el producto escalar $< >: L^2(\Omega) \times L^2(\Omega) \to R$ de la forma siguiente:

tal que $a(v,v) > \alpha|v|^2_{H_0^1(\Omega)}$ definida por:

$$a(u,v) = \int_0^{\ell} \nu \frac{du}{dx} \frac{dv}{dx}\, dx.$$

Si w es solución del problema P_{EHG}, esto no implica que w será solución de P_{EH}; ya que $w \in H_0^1(\Omega)$ y $H_0^1(\Omega) \not\subset C^2(\Omega)$; por lo que w sólo satisface la ecuación P_{EH} en términos de distribuciones o funciones generalizadas [2]. Para diferenciar a ambas soluciones, se llama a la solución del problema P_{EH} la solución clásica y a la de P_{EHG} la solución débil o generalizada.

II. Algunos métodos numéricos para aproximar la solución del problema P_{EH} y P_{EHG}.

Consideremos de nuevo el problema P_{EH} y definamos una función llamada Residuo $R: V \to C^0(|0,\ell|)$

$$R(u) = Lu + g(u) - f.$$

Si w es solución de P_{EH} entonces w satisface:

$$w \in W$$
$$R(w) = 0$$
$$w(0) = w(\ell) = 0.$$

Hay muchos métodos para construir una aproximación a la solución del problema P_{EH}. Algunos de estos métodos se conocen con el nombre de métodos de Residuos Pesados y fueron introducidos en los cinco primeros capítulos de las notas del curso de Análisis Numérico [3]. Veamos que obtenemos al aplicarlos a P_{EH}.

Método de Colocación. Sea W_h un subespacio de dimensión finita de W y sea $\{v_1^h,\ldots,v_n^h\}$ una base de W_h. Sean $x_1,\ldots,x_n \in \Omega$. El método de colocación busca una función $w_h \in W_h$ que satisfaga:

$$(1) \qquad \begin{aligned} &R(w_h)(x_j) = 0 \qquad j = 1,\ldots,n \\ &w_h(0) = w_h(\ell) = 0 \end{aligned}$$

Como $w_h \in W_h$ tenemos que $w_h(x) = \sum_{i=1}^{n} \alpha_i v_i^h(x)$ y sustituyendo en (1) obtenemos:

$$\sum_{i=1}^{n} Lv_i^h(x_j)\alpha_i + g\left(\sum_{i=1}^{n} \alpha_i v_i^h + u_0\right)(x_j) = f(x_j) \qquad j = 1,\ldots,n$$

Sean A la matriz $n \times n$ definida por $A_{ji} = Lv_i^h(x_j)$, $\vec{b}$ el vector $b_j = f(x_j)$ y $\vec{\alpha} = (\alpha_1,\ldots,\alpha_n)$; entonces resolver (1) se reduce a resolver el siguiente sistema de ecuaciones no lineales:

$$A\vec{\alpha} + \vec{g}(\vec{\alpha}) = \vec{b}.$$

Método de Subdominios. Sean n un natural, $h = \frac{\ell}{n}$, $a_0 = 0$. Definamos los puntos $a_i = a_0 + ih$, $1 < i < n$. Entonces $a_n = \ell$ y $a_i \in \Omega$ para $i = 0,\ldots,n$. La Partición P_h de Ω en intervalos cerrados $|a_i,a_{i+1}|$ $0 < i < n-1$ satisface:

$$\bar{\Omega} = \bigcup_{i=0}^{n-1} |a_i,a_{i+1}| \quad y \quad <a_i,a_{i+1}> \cap <a_j,a_{j+1}> \text{ si } i \neq j$$

Sea

$$\chi_j(x) = \begin{cases} 1 & \text{si } x \ |a_j,a_{j+1}| \\ 0 & \text{en otro lado.} \end{cases}$$

El método de subdominios busca una función $w_h \in W_h$ tal que:

$$\int_0^{\ell} R(w_h)\chi_j(x)dx = 0 \qquad j = 1,\ldots,n-1$$

(2)

$$w_h(0) = w_h(\ell) = 0.$$

De nuevo, como $w_h \in W_h$ debe tenerse que $w_h(x) = \sum_{i=1}^{n} \alpha_i v_i^h(x)$ y sustituyendo en (2) obtenemos:

$$\sum_{i=1}^{n} \alpha_i \int_0^{\ell} Lv_i^h(x)\chi_j(x)dx + \int_0^{\ell} g\left(\sum_{i=1}^{n} \alpha_i v_i^h(x) + u_0\right)\chi_j(x)dx =$$

$$= \int_0^{\ell} f(x)\chi_j(x)dx.$$

Si

$$A_{ji} = \int_0^{\ell} Lv_i^h(x)\chi_j(x)dx \quad y \quad b_j = \int_0^{\ell} f(x)\chi_j(x)dx$$

Resolver (2) se reduce a resolver:

$$A\vec{\alpha} + \vec{\gamma}(\vec{\alpha}) = \vec{b}$$

donde $\gamma_j(\vec{\alpha}) = \int_0^{\ell} g\left(\sum_{i=1}^{n} \alpha_i v_i^h(x) + u_0(x)\right)\chi_j(x)dx.$

<u>Método de Galerkin</u>. Sea $\{v_i^h\}_{i=1}^n$ una base de V_h. El método de Galerkin nos da una forma de construir una $w_h \in W_h$ que satisfaga:

$$\int_0^{\ell} R(w_h)v_j^h dx = 0 \qquad j = 1,\ldots,n$$

(3)

$$w_h(0) = w_h(\ell) = 0$$

Como $w_h \in W_h$ existen escalares $\alpha_1,\ldots,\alpha_n$ tales que $w_h(x) = \sum_{i=1}^{n} \alpha_i v_i^h(x)$ y sustituyendo en (3) obtenemos:

$$\sum_{i=1}^{n} \alpha_i \int_0^{\ell} Lv_i^h v_j^h dx + \int_0^{\ell} g\left(\sum_{i=1}^{n} \alpha_i v_i^h + u_0 \right) v_j^h dx$$

(4)

$$= \int_0^{\ell} f v_j^h dx$$

si

$$A_{ji} = \int_0^{\ell} Lv_i^h v_j^h dx, \quad bj = \int_0^{\ell} f v_j^h dx$$

y

$$\gamma_j(\vec{\alpha}) = \int_0^{\ell} g\left(\sum_{i=1}^{n} \alpha_i v_i^h + u_0 \right) v_j^h dx.$$

La ecuación (3) se reduce a resolver:

$$A\vec{\alpha} + \vec{\gamma}(\vec{\alpha}) = \vec{b}$$

Así pués, los tres métodos anteriores para encontrar una solución aproximada de P_{EH}, nos llevan a resolver un sistema de ecuaciones no lineales. Observemos que $\{v_i^h\}_{i=1}^n$ deberá estar contenida en $C^2(\Omega)$ para que $W_h \in W$. Una forma de lograrlo es construir la base $\{v_i^h\}_{i=1}^n$ utilizando "splines cúbicos" [4].

Mientras mayor es el grado de los polinomios que estamos utilizando, mayor es el número de elementos de nuestra matriz A y de los vectores $\vec{\alpha}$, $\vec{b}$ y $\vec{\gamma}$. Esto aumenta los

requerimientos de memoria y de tiempo de computadora [4]. El método de Galerkin nos permite construir una solución aproximada a partir de polinomios de primer grado si en lugar de aproximar la solución de P_{EH} lo hacemos para P_{EHG}.

Observemos que al aplicar integración por partes a la ecuación (4) obtenemos:

$$\int_0^{\ell} \nu \frac{dw_h}{dx} \frac{dv_j^h}{dx} dx + \int_0^{\ell} g(w_h + u_0) v_j^h dx =$$

$$\int_0^{\ell} f v_j^h dx + \left| \frac{dw_h}{dx} (x) v_j^h(x) \right|_0^{\ell}.$$

Si construimos nuestro espacio W_h de tal forma que éste sea un subespacio de $H_0^1(\Omega)$ obtendremos:

$$P_h \begin{cases} w_h \in W_h \quad \text{tal que} \\ \\ a(w_h, v_h) + \langle g(w_h + u_0), v_h \rangle = \langle f, v_h \rangle \text{ toda } w_h \in W_h \end{cases}$$

Al resolver P_h estamos resolviendo P_{EHG} en un subespacio W_h de W de dimensión finita.

III. Construcción del espacio $W_h \in H_0^1(\Omega)$.

Consideremos la partición P_h que fue introducida en el método de subdominios. Esta partición depende de h, por

lo que al reducir o aumentar su valor, generaríamos nuevas particiones. Definamos las siguientes funciones:

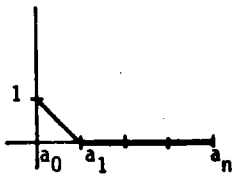
$$v_h^0(x) = \begin{cases} \dfrac{x-a_0}{a_0-a_1} & \text{si} \quad x \in |a_0, a_1| \\ 0 \text{ en otro lado} \end{cases}$$

$1 < i < n-1.$

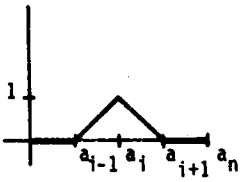
$$v_h^i(x) = \begin{cases} \dfrac{x-a_{i-1}}{a_i-a_{i-1}} & \text{si} \quad x \in |a_{i-1}, a_i| \\ \dfrac{x-a_{i+1}}{a_i-a_{i+1}} & \text{si} \quad x \in |a_i, a_{i+1}| \\ 0 \text{ en otro lado} \end{cases}$$

$$v_h^n(x) = \begin{cases} \dfrac{x-a_{n-1}}{a_n-a_{n-1}} & x \in |a_{n-1}, a_n| \\ 0 \text{ en otro lado.} \end{cases}$$

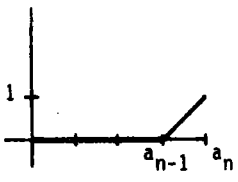
La gráfica de estas funciones es de la forma siguiente:



$v_h^0(x)$



$v_h^i(x)$



$v_h^n(x)$

Observemos que $v_i^h \in C^0(\Omega)$ para $0 < i < n$ y que la restricción de v_i^h a cada intervalo $|a_i, a_{i+1}|$ es un polinomio de primer grado; $v_i^h \in H^1(\Omega)$ para $0 < i < n$ porque para cada intervalo $|a_i, a_{i+1}|$ $v_i^h \in H^1(|a_i, a_{i+1}|)$ y además $v_i^h \in C^0(\Omega)$.

Sea W_h el espacio generado por $\{v_0^h, \ldots, v_n^h\}$. Entonces este conjunto es una base de W_h por ser sus elementos linealmente independientes. Sea U_h el espacio generado por $\{v_1^h, \ldots, v_{n-1}^h\}$; éste es un subespacio de W_h en el que para toda $v_h \in U_h$ $v_h(a) = v_h(\ell) = 0$. Por lo que $W_h \in H^1(\Omega)$ y $U_h \in H_0^1(\Omega)$.

Resolver el problema P_h se reduce a buscar $w_h \in U_h$ tal que si $w_h(x) = \sum_{j=1}^{n-1} \alpha_j v_i^h(x)$, los coeficientes $\{\alpha_j\}_{j=1}^{n-1}$ satisfagan:

$$\sum_{j=1}^{n-1} \alpha_j a(v_j^h, v_i^h) + \left\langle g\left(\sum_{j=1}^{n-1} \alpha_j v_j^h + u_0 \right), v_i^h \right\rangle = \langle f, v_i^h \rangle$$

$$1 < i < n-1.$$

Sea $A_{ij} = a(v_i^h, v_j^h)$ y $b_i = \langle f, v_i^h \rangle$. Por cálculo directo se obtiene que A es una matriz tridiagonal, simétrica y positiva definida.

$$A = \begin{vmatrix} \frac{2\nu}{h} & \frac{-\nu}{h} & 0 & 0 & \cdots & 0 \\ \frac{-\nu}{h} & \frac{2\nu}{h} & \frac{-\nu}{h} & 0 & \cdots & 0 \\ 0 & \frac{-\nu}{h} & \frac{2\nu}{h} & \frac{-\nu}{h} & 0 \cdots & 0 \\ & & & & & \\ 0 & & & & 0 \; \frac{-\nu}{h} & \frac{2\nu}{h} \end{vmatrix}$$

Para calcular el vector $\vec{b}$ debemos construir la función u_0, ya que $f = -Lu_0$. Sea $u_0(x) = av_0^h(x) + bv_n^h(x)$. Claramente $u_0 \in W_h \subset H^1(\Omega)$. Entonces u_0 satisface las condiciones a la frontera: $u_0(0) = a$, $u_0(\ell) = b$.

$$b_i = \langle f, v_i^h \rangle = \int_0^\ell -Lu_0 v_i^h dx = \int_0^\ell \nu \frac{du_0}{dx} \frac{dv_i^h}{dx} dx$$

$$^T\vec{b} = \left| \frac{-\nu a}{h}, 0, \ldots\ldots, 0, \frac{-\nu b}{h} \right|$$

<u>Cálculo del vector</u> $\vec{\gamma}$. Dado que

$$\gamma_i = \int^\ell g \left(\sum_{j=1}^{n-1} \alpha_j v_j^h + av_0^h + bv_n^h \right) v_i^h dx$$

tenemos que

$$A = \begin{vmatrix} \frac{2\nu}{h} & \frac{-\nu}{h} & 0 & 0 & \cdots & 0 \\ \frac{-\nu}{h} & \frac{2\nu}{h} & \frac{-\nu}{h} & 0 & \cdots & 0 \\ 0 & \frac{-\nu}{h} & \frac{2\nu}{h} & \frac{-\nu}{h} & 0 \cdots & 0 \\ & & & & & \\ 0 & \cdots\cdots\cdots & & & 0 & \frac{-\nu}{h} \quad \frac{2\nu}{h} \end{vmatrix}$$

Para calcular el vector $\vec{b}$ debemos construir la función u_0, ya que $f = -Lu_0$. Sea $u_0(x) = av_0^h(x) + bv_n^h(x)$. Claramente $u_0 \in W_h \subset H^1(\Omega)$. Entonces u_0 satisface las condiciones a la frontera: $u_0(0) = a$, $u_0(\ell) = b$.

$$b_i = \langle f, v_i^h \rangle = \int_0^\ell -Lu_0 v_i^h dx = \int_0^\ell \nu \frac{du_0}{dx} \frac{dv_i^h}{dx} dx$$

$$^{\tau}\vec{b} = \left| \frac{-\nu a}{h}, 0, \ldots\ldots, 0, \frac{-\nu b}{h} \right|$$

<u>Cálculo del vector</u> $\vec{\gamma}$. Dado que

$$\gamma_i = \int^\ell g \left(\sum_{j=1}^{n-1} \alpha_j v_j^h + av_0^h + bv_n^h \right) v_i^h dx$$

tenemos que

$$(5)\qquad \gamma_i(\vec{\alpha}) = \begin{cases} \frac{1}{6}\left|\alpha_{i+1}^2 + \alpha_i\alpha_{i+1} - \alpha_{i-1}\alpha_i - \alpha_{i-1}^2\right| & 2 < 1 < n-2 \\ \frac{1}{6}\left|\alpha_2^2 + \alpha_1\alpha_2 - a\alpha_1 - a^2\right| & i = 1 \\ \frac{1}{6}\left|b^2 + b\alpha_{n-1} - \alpha_{n-1}\alpha_{n-2} - \alpha_{n-2}^2\right| & i = n-1. \end{cases}$$

Resolver P_h se reduce a resolver el siguiente sistema de ecuaciones no lineales:

$$A\vec{\alpha} + \vec{\gamma}(\vec{\alpha}) = \vec{b}.$$

IV. Resolución de la ecuación $A\vec{\alpha} + \vec{\gamma}(\vec{\alpha}) = \vec{b}$.

Dado que la ecuación es no lineal utilizaremos alguno de los métodos vistos en el Capítulo 8 de [3]. Por ejemplo el de iteraciones sucesivas:

1) Sea $\vec{\alpha}_0$ la solución del sistema lineal:

$$A\vec{\alpha} = \vec{b}$$

2) Calcúlese $\vec{\alpha}_{k+1}$ resolviendo

$$A\vec{\alpha}_{k+1} = b - \vec{\gamma}(\vec{\alpha}_k).$$

En las notas anterioremente citadas se demuestra que este

método convergerá si $\|A^{-1}D\gamma(\vec{\alpha})\| < 1$ donde $D\gamma(\vec{\alpha})$ es la matríz jacobiana de $\vec{\gamma}$ y $\|\ \|$ es la norma espectral.

Cálculo de $D\gamma(\vec{\alpha})$. Utilizando (5) observamos que la matríz $D\gamma(\vec{\alpha})$ tiene la siguiente forma:

$$\partial_i \vec{\gamma}_j(\vec{\alpha}) = (D\gamma(\vec{\alpha}))_{ij} = \begin{cases} \frac{2\alpha_{i+1} + \alpha_1}{6} & \text{si } i = j+1 \\ \frac{\alpha_{i+1} - \alpha_{i-1}}{6} & \text{si } i = j \\ \frac{-\alpha_i - 2\alpha_{i-1}}{6} & \text{si } i = j-1 \\ 0 & \text{en otra parte.} \end{cases}$$

$D\gamma(\vec{\alpha})$ es una matríz tridiagonal no simétrica.

Otro método que puede aplicarse es el de Newton Raphson:

1) Calcule $\vec{\alpha}_0$ como solución de

$$A\vec{\alpha} = \vec{b}$$

2) Calcule $\vec{\alpha}_{k+1}$ resolviendo el siguiente sistema

$$[A + D\gamma(\vec{\alpha}_k)][\vec{\alpha}_{k+1} - \vec{\alpha}_k] = A\vec{\alpha}_k + \vec{\gamma}(\vec{\alpha}_k) - \vec{b}$$

Si el método de Newton Raphson converge para el valor inicial $\vec{\alpha}_0$ así escogido lo hará cuadráticamente, compen-

sando asi el costo de calcular y almacenar en cada iteración $D\vec{Y}(\vec{\alpha}_k)$.

V. Resolución de la ecuación de Burgers no estacionaria.

Si consideramos ahora que $\frac{\partial u}{\partial t} \neq 0$ en el problema P, la ecuación a resolver es la siguiente:

$$P \quad \begin{cases} \frac{\partial u}{\partial t} + g(u) = Lu \quad \text{en} \quad <0,\ell> \times <0,t_0> \\ u(0,t) = a(t) \qquad u(\ell,t) = b(t) \\ u(x,0) = r_0(x) \end{cases}$$

Sea $V = C^2(<0,\ell>) \times C^1(<0,t_0>)$. Sea $u_0 \in V$ tal que $u_0(0,t) = a(t)$ y $u_0(\ell,t) = b(t)$. Si resolvemos el problema

$$P_H \quad \begin{cases} w \in V \quad \text{tal que} \\ \frac{\partial}{\partial t}(w+u_0) + g(w+u_0) + L(u+u_0) = 0 \\ w(0,t) = w(\ell,t) = 0 \\ w(x,0) = r_0(x) - u_0(t,0) \end{cases}$$

La solución del problema P será de la forma

$(w+u_0)$ (x,t). De nuevo, u_0 se construye igual que en III, es decir

$$u_0(x,t) = a(t)v_0^h(x) + b(t)v_n^h(x). \tag{6}$$

Sea w solución de P_H por lo que para toda $v \in H_0^1(\Omega)$ tenemos:

$$P_{HG} \begin{cases} \left\langle \frac{\partial w}{\partial t}, v \right\rangle + \langle g(w+u_0), v\rangle + a(w+u_0, v) = 0 \\ \\ \langle w, v\rangle_{t=0} = \langle r_0 - u_0, v\rangle_{t=0} \end{cases}$$

Para aproximar la solución de este problema, podemos aplicar, en cada instante t, el método de Galerkin que desarrollamos en las secciones anteriores. Consideremos que la solución aproximada w_h está en $U_h \times H^1(<0,t_0>)$ y es de la forma:

$$w_h(x,t) = \sum_{i=1}^{n-1} \alpha_i(t)v_i^h(x). \tag{7}$$

$u_0(x,t)$ definida por (6) satisface: $u_0 \in W_h \times H^1(<0,t_0>)$. En este caso para cada $t \in <0,t_0>$ tendremos un vector de coeficientes $\vec{\alpha}$.

El resolver P_{HG} en el espacio U_h equivale a resolver el siguiente sistema de ecuaciones:

$$< \frac{\partial w_h}{\partial t}, v_i^h > + <g(w_h+u_0), v_i^h> + a(w_h+u_0, v_i^h) = 0$$

(8)

$$<w_h, v_i^h>_{t=0} = <r_0-u_0, v_i^h>_{t=0}$$

donde $\{v_i^h\}_{i=1}^{n-1}$ es la base de U_h construida en la Sec. III.

Substituyendo (7) y (6) en (8) obtenemos:

$$\sum_{j=1}^{n-1} <v_i^h, v_j^h> \alpha_j(t) + <g\left(\sum_{j=1}^{n-1} \alpha_j(t) v_j^h + a(t) v_0^h + b(t) v_n^h\right), v_i^h>$$

$$+ \sum_{j=1}^{n-1} a(v_i^h, v_j^h)\ \alpha_j(t) = <f, v_i^h> - <a(t) v_0^h + b(t) v_n^h, v_i^h>$$

$$\sum_{j=1}^{n-1} <v_i^h, v_j^h> \alpha_j(0) = <r_0(x) - a(0) v_0^h - b(0) v_n^h, v_i^h>$$

Si

$$b_i = <f, v_i^h> - <\dot{a}(t) v_0^h + \dot{b}(t) v_n^h, v_i^h>$$

$$A_{ij} = a(v_i^h, v_j^h) \qquad C_{ij} = <v_i^h, v_j^h>$$

El resolver (8) equivale a resolver el siguiente sistema de ecuaciones diferenciales ordinarias.

$$C\dot{\vec{\alpha}}(t) + A\vec{\alpha}(t) + \vec{\gamma}(\vec{\alpha}(t)) = \vec{b}$$

(9)

$$C\vec{\alpha}(0) = \vec{d}$$

donde $d_i = \langle r_0(x) - a(0)v_0^h - b(0)v_n^h, v_i^h \rangle$.

En el Capítulo 9 de [3] se describen varios métodos numéricos para resolver este problema.

Por ejemplo, el método de Euler para aproximar la solución de (9) conduce al siguiente algoritmo.

Sea $\ell = \frac{t_0}{m}$ para alguna $m \in \mathbb{N}$. Denotaremos $\vec{\alpha}_i$ a la aproximación de $\vec{\alpha}$ en $t = i\ell$.

1) Si $\vec{\alpha}_0$ es la solución de $C\vec{\alpha}(0) = \vec{d}$.

2) Obtenga $\vec{\alpha}_{i+1}$ resolviendo el siguiente sistema:

$$C\vec{\alpha}_{i+1} = C\vec{\alpha}_i + \ell[b - A\vec{\alpha}_i - \vec{\gamma}(\vec{\alpha}_i)]$$

Observe que utilizando este método nos evitamos resolver un sistema de ecuaciones no lineales. Desgraciadamente este método es poco recomendable para resolver problemas no lineales. Si utilizamos un esquema implícito como Crank-Nicolson tendremos necesariamente que resolver, en cada paso, un sistema de ecuaciones no lineales, logrando, a cambio, una mejor precisión. El algoritmo es el siguiente:

1) Sea $\vec{\alpha}_0$ solución de $C\vec{\alpha}(0) = \vec{d}$.

2) $\vec{\alpha}_{i+1}$ es solución de

$$\frac{1}{\ell} C[\vec{\alpha}_{i+1} - \vec{\alpha}_i] + \frac{A}{2}[\vec{\alpha}_{i+1} + \vec{\alpha}_i] + \vec{\gamma}\left[\frac{\vec{\alpha}_{i+1}+\vec{\alpha}_i}{2}\right] = \left(i+\frac{1}{2}\right)\ell\vec{b}$$

Dado que el vector $\vec{\alpha}_i$ se conoce, esto equivale a resolver:

$$F(\vec{\alpha}_{i+1}) = \frac{1}{\ell} B\vec{\alpha}_{i+1} + \frac{1}{2} A\vec{\alpha}_{i+1} + \vec{\gamma}\left[\frac{\vec{\alpha}_{i+1}+\vec{\alpha}_i}{2}\right] = \vec{e}\,.$$

Nótese que F es no lineal, por lo que resolver la ecuación

$$F(\vec{\alpha}) = \vec{e}$$

requiere utilizar alguno de los métodos iterativos a los que ya hicimos referencia en la Sección IV.

VI. Conclusiones.

El método de Galerkin ofrece muchas ventajas al ser aplicado a un problema del tipo de la ecuación de Burgers. Desde el punto de vista teórico resolver el problema P_{HG} es mucho más sencillo que resolver el problema P: desaparece la derivada parcial de segundo orden, permitiendonos aproximar la solución con funciones parecidas a polinomios de primer grado, y el espacio de funciones donde buscamos la solución es mucho menos restrictivo: $H^1(\Omega)$ en lugar de $C^2(\Omega)$. Desde el punto de vista numérico hay varias ventajas:

Siendo los coeficientes $\vec{\alpha}(t)$ los que dan la variación de la solución aproximada $w_h(x,t)$ en el tiempo t, el cálculo de los elementos de la base $\{v_i^h\}_{i=1}^{n-1}$ como el de las matrices A, B se hace una sóla vez; esto permite un ahorro considerable de tiempo máquina. Así también que el que las funciones base v_i^h se anulen fuera del intervalo $[a_{i-1}, a_{i+1}]$ implica que la estructura de las matrices A, C y $D\vec{\gamma}(\vec{\alpha})$ sea bandeada. Esto se traduce en menos memoria y menos tiempo de cálculo al resolver $F(\vec{\alpha}_{i+1}) = \vec{e}$. Aún más, la naturaleza del problema P_{HG} hace que A y B sean simétricas y positivas definidas.

Por último el fácil cálculo de $D\vec{\gamma}(\vec{\alpha})$ permite utilizar Newton Raphson y por lo consiguiente un método implícito para resolver la ecuación diferencial (9), que es mucho más exacto que uno de tipo Euler o Runge-Kutta.

BIBLIOGRAFIA

[1] Enzo Levi; Mecánica de Fluidos. Facultad de Ingeniería (UNAM), 1965.

[2] Saúl Hahn Goldberg; Distribuciones y Transformadas de Fourier. Notas del III Coloquio del Depto. de Matemáticas (CIEA-IPN), 1983.

[3] Diego Bricio Hernández; Análisis Numérico: Aproximación y Algorítmica. Notas del III Coloquio del Depto. de Ma-

temáticas (CIEA-IPN), 1983.

[4] Prenter, P.; Splines and Variational Methods. J. Wiley and Sons, New York, 1975.

UNA INTRODUCCION A LOS PRINCIPIOS VARIACIONALES Y AL METODO DE RITZ

Hugo Martínez*

Resumen

Una gran cantidad de problemas prácticos pueden modelarse ya sea por una funcional, (la cual en algunos casos puede ser interpretada en términos de la energía del sistema bajo consideración) como por ejemplo, el principio de mínima acción o por ecuaciones diferenciales, como por ejemplo, la segunda ley de Newton. Si el problema es modelado por una funcional y la solución es uno de los puntos críticos de esta funcional podríamos entonces utilizar el método de Ritz (el cual describiremos posteriormente) para encontrar aproximaciones a dicha solución. En cambio si el problema es modelado por ecuaciones diferenciales y deseamos utilizar el método de Ritz para encontrar aproximaciones a la solución de dicho problema, entonces, lo que haremos es buscar un principio variacional. Entendiendo por esto, una proposición que establece la equivalencia entre las soluciones de una basicación y los puntos críticos de una funcional asociada, es decir, una funcional tal que las soluciones de las ecuaciones coincidan con los puntos críticos de la funcional. Para llevar a cabo la cons-

* Departamento de Matemáticas, Universidad Autónoma Metropolitana, Unidad Iztapalapa

trucción de esta funcional asociada, lo que haremos es buscar un principio variacional; entendiendo por esto, una proposición que establece la equivalencia entre las soluciones de una ecuación y los puntos críticos de una cierta funcional asociada. Para llevar a cabo la construcción de esta funcional asociada, primero analizaremos problemas definidos en espacios de dimensión finita y posteriormente consideraremos problemas definidos en espacios de dimensión infinita. En este estudio sólo se considerarán problemas lineales.

Nota. El método de Ritz se podría aplicar si se tiene una formulación débil del problema. Esta formulación puede obtenerse de varias maneras. En estas notas, esencialmente lo que haremos, es obtener una formulación débil via los principios variacionales.

PROBLEMA 1.

Consideremos la ecuación

$$ax = b \qquad a,x,b \in \mathbb{R} \tag{1}$$

Según lo que planteamos, deseamos encontrar una funcional f tal que sus puntos críticos coincidan con las soluciones de (1), es decir, una f tal que

$$\nabla f(x) = ax - b$$

En este caso, es claro que integrando encontramos que

$$f(x) = \frac{1}{2} ax^2 - bx$$

tiene las propiedades deseadas.

Obsérvese que si la funcional f es convexa $(a > 0)$ se tiene un principio variacional de mínimo y si la funcional es cóncava $(a < 0)$ se tiene un principio variacional de máximo.

PROBLEMA 2.

Consideremos la ecuación

$$Ax = b;\ A \text{ una matriz } n \times n;\quad x, b \in \mathbb{R}^n$$

deseamos encontrar una f tal que

$$\nabla f(x) = Ax - b \tag{2}$$

Si definimos $F: \mathbb{R}^n \to \mathbb{R}^n$ por $F(x) = Ax - b$, entonces el problema de saber si existe una f que satisfaga (2) es equivalente a preguntarse si el campo F es potencial y éste será el caso si y sólo si la derivada del campo F forma una matriz simétrica (véase [1]).

Supongamos que A es simétrica, por lo anterior sabemos que existe una función potencial y un procedimiento para encontrarla consiste en integrar parcialmente las siguientes ecuaciones

$$\frac{\partial f}{\partial x_1} = a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n - b_1$$

$$\cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot$$

$$\frac{\partial f}{\partial x_n} = a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n - b_n$$

De donde obtenemos

$$f(x_1,\ldots,x_n) = \frac{1}{2} \langle Ax,x\rangle - \langle b,x\rangle$$

Aquí

$$\langle x,y\rangle = x_1y_1 + x_2y_2 + \ldots + x_ny_n$$

Si la funcional f es convexa (A definida positiva) se tiene un principio variacional de mínimo y si la funcional es cóncava (A definida negativa) se tiene un principio variacional de máximo.

En la construcción de funcionales asociadas a problemas con valores en la frontera en ecuaciones diferenciales procederemos de la siguiente manera: Presentaremos unas funcionales y definiremos su gradiente y posteriormente trataremos de identificar los problemas dados con estos gradientes.

Sea $f:C'[a,b] \to \mathbb{R}$ definida por:

$$f(\phi) = \int_a^b \tilde{f}(\phi(t),\phi'(t),t)dt. \qquad (3)$$

donde $\tilde{f}:\mathbb{R}^3 \to \mathbb{R}$ es una función de clase C^2.

Puesto que la diferencial de Fréchet de f en ϕ (la

que denotamos por df_ϕ) calculada en $M = \{h \in C'[a,b] \mid h(a) = h(b) = 0\}$ está dada por:

$$df_\phi(h) = \int_a^b \left[\frac{\partial \tilde{f}}{\partial x} (\phi(t),\phi'(t),t) - \frac{d}{dt} \left(\frac{\partial \tilde{f}}{\partial y} (\phi(t),\phi'(t),t) \right) \right] h(t)dt$$

Por analogía con la diferencial de una función de varias variables y tomando en cuenta que el producto escalar para funciones se da por medio de una integral, resulta natural definir el gradiente de f en ϕ por:

$$\nabla f_\phi = \frac{\partial \tilde{f}}{\partial x} (\phi(t),\phi'(t),t) - \frac{d}{dt} \left(\frac{\partial \tilde{f}}{\partial y} (\phi(t),\phi'(t),t) \right) \qquad (4)$$

véase [2], donde $x = \phi(t)$, $y = \phi'(t)$.

Nota. El dominio de definición del gradiente al igual que la expresión misma del gradiente dependen del espacio en el cual el dominio de la funcional está contenido. Aquí suponemos que $C'[a,b] \subset L^2[a,b]$.

PROBLEMA 3

Consideremos el siguiente problema con valores en la frontera en ecuaciones diferenciales ordinarias

$$A(\phi) = -\frac{d^2\phi(t)}{dt^2} - \phi(t) = t^2$$

$$\phi(0) = 0$$

$$\phi(1) = 0$$

Puesto que la expresión (4) en general es una expresión de segundo orden, se espera que podamos encontrar una f de la forma (3) tal que

$$\nabla f_{\phi} = -\frac{d^2\phi(t)}{dt^2}\,\phi(t) - t^2.$$

Un procedimiento para encontrar una f (de donde su existencia) es análogo al del Ejemplo 2.

$$\nabla f_{\phi} = \frac{\partial \tilde{f}}{\partial x}(\phi(t),\phi'(t),t) - \frac{d}{dt}\left(\frac{\partial \tilde{f}}{\partial y}(\phi(t),\phi'(t),t)\right) = -\phi(t) - t^2 - \frac{d}{dt}(\phi'(t))$$

de donde

$$\frac{\partial \tilde{f}}{\partial y}(\phi(t),\phi'(t),t) = \phi'(t) = y$$

$$\frac{\partial \tilde{f}}{\partial x}(\phi(t),\phi'(t),t) = -\phi(t) - t^2 = -x - t^2$$

que integrando se obtiene

$$\tilde{f}(x,y,t) = \frac{y^2}{2} - \frac{x^2}{2} - xt^2$$

Por tanto, una f está dada por:

$$f(\phi) = \int_0^1 \left[\frac{\phi'(t)^2}{2} - \frac{\phi(t)^2}{2} - \phi(t)t^2\right] dt$$

Si ϕ es de clase C^2 entonces nótese que

$$f(\phi) = \frac{1}{2}\langle A\phi,\phi\rangle - \langle b,\phi\rangle$$

donde

$$\langle f,g\rangle = \int_0^1 f(t)g(t)dt$$

y

$$b(t) = t^2.$$

PROBLEMA 4

Consideremos el siguiente problema con condiciones de frontera no-homogéneas

$$\begin{aligned} t^2 \frac{d^2\psi}{dt^2} + t \frac{d\psi}{dt} + (t^2-1)\psi &= 0 \\ \psi(1) &= 2 \\ \psi(2) &= 4 \end{aligned} \tag{5}$$

haciendo el cambio

$$\psi = \phi+2t$$

el problema (5) se transforma en un problema con condiciones de fontera homogéneas.

$$\begin{aligned} t\phi''(t) + \phi'(t) + \frac{t^2-1}{t}\phi(t) + 2t^2 &= 0 \\ \phi(1) &= 0 \\ \phi(2) &= 0 \end{aligned} \tag{6}$$

Ahora encontramos una funcional asociada para el

problema (6)

$$\nabla f_{\phi} = \frac{\partial \tilde{f}}{\partial x} - \frac{d}{dt}\left(\frac{\partial \tilde{f}}{\partial y}\right) = -\frac{t^2-1}{t}\phi - 2t^2 - \frac{d}{dt}(t\phi')$$

de donde

$$\frac{\partial \tilde{f}}{\partial y} = t\phi' = ty$$

$$\frac{\partial \tilde{f}}{\partial x} = -\frac{t^2-1}{t}\phi - 2t^2 = -\frac{t^2-1}{t}x - 2t^2$$

que integrando se obtiene

$$\tilde{f}(x,y,t) = \frac{ty^2}{2} - \frac{t^2-1}{2t}x^2 - 2t^2x .$$

Por tanto, una f está dada por:

$$f(\phi) = \int_1^2 \left[\frac{t\phi'(t)^2}{2} - \frac{t^2-1}{2t}\phi(t)^2 - 2t^2\phi(t)\right] dt .$$

-o-

Ahora sea $f:C'(\Omega) \to \mathbb{R}$ definida por:

$$f(\phi) = \int_{\Omega} \tilde{f}\left(x_1, x_2, \phi(x_1,x_2), \frac{\partial \phi}{\partial x_1}, \frac{\partial \phi}{\partial x_2}\right) dx_1, dx_2$$

donde $\tilde{f}:\mathbb{R}^5 \to \mathbb{R}$ es una función de clase C^2 y Ω es un conjunto simplemente conexo, cerrado y acotado con frontera

lisa en $\mathbb{R}^2$.

Puesto que la diferencial de Fréchet de f en ϕ calculada en $M = \{ h \in C'(\Omega) ; h\big|_{\partial\Omega} = 0 \}$, está dada por:

$$df_{\phi}(h) = \int_{\Omega} \left[\frac{\partial \tilde{f}}{\partial y} - \sum_{i=1}^{2} \frac{\partial}{\partial x_i} \left(\frac{\partial \tilde{f}}{\partial z_i} \right) \right] h(x_1, x_2) dx_1 dx_2$$

resulta natural definir el gradiente de f en ϕ por:

$$\nabla f_{\phi} = \frac{\partial \tilde{f}}{\partial y} - \sum_{i=1}^{2} \frac{\partial}{\partial x_i} \left(\frac{\partial \tilde{f}}{\partial z_i} \right)$$

donde

$$x = (x_1, x_2)$$

$$y = \phi(x_1, x_2)$$

$$z = \left(\frac{\partial \phi}{\partial x_1} , \frac{\partial \phi}{\partial x_2} \right)$$

PROBLEMA 5.

Consideremos el siguiente problema con valores en la frontera, llamado problema de Dirichlet para la ecuación de Poisson

$$A(\phi) = - \frac{\partial^2 \phi}{\partial x_1} - \frac{\partial^2 \phi}{\partial x_2} = b(x_1, x_2) \quad \text{en } \Omega$$

$$\phi = 0 \quad \text{en} \quad \partial\Omega$$

Procediendo como en el Problema 3, no es difícil verificar que una funcional asociada a este problema está dado por:

$$f(\phi) = \int_{\Omega} \left[\frac{1}{2} \left(\frac{\partial \phi}{\partial x_1} \right)^2 + \frac{1}{2} \left(\frac{\partial \phi}{\partial} \right)^2 - b(x_1, x_2)\phi(x_1, x_2) \right] dx_1\, dx_2$$

Además, si $\bar{\phi}$ es de clase C^2 entonces

$$f(\phi) = \frac{1}{2} \langle A\phi, \phi \rangle - \langle b, \phi \rangle.$$

En general, si:

a) D denota un subespacio lineal denso de un espacio de Hilbert real H (con producto interior $\langle , \rangle$) (véase [2]).

b) $A: D \to H$ es un operador definido positivo, es decir,

   i) Es simétrico, es decir, $\langle Ax, y \rangle = \langle x, Ay \rangle \ \forall\ x, y \in D$

   ii) Existe $k^2 > 0$ tal que $\langle Ax, x \rangle \geq k^2 |x|^2 \ \forall\ x \in D$

c) Para cada b fija en H, consideramos la funcinal $f: D \to \mathbb{R}$ definida por:

$$f(x) = \frac{1}{2} \langle Ax, x \rangle - \langle b, x \rangle$$

se tiene el siguiente principio vatiacional.

Resultado 1. Si la ecuación $Ax = b$ tiene una solución en D, esta solución minimiza la funcional f; inversamente, si existe un elemento en D que minimiza la funcional f,

este elemento satisface $Ax = b$.

Nota. Puesto que A es definido positivo, la ecuación $Ax = b$ tiene a lo más una solución.

Demostración. Supongamos que x_0 es solución de $Ax = b$ de donde

$$2f(x) = \langle Ax,x\rangle - 2\langle x,Ax_0\rangle = \langle A(x-x_0),x-x_0\rangle - \langle Ax_0,x_0\rangle.$$

Por lo tanto, f alcanza su mínimo en $x = x_0$.

Inversamente, supóngase que f alcanza su mínimo en $x = x_0$, de donde la función $f(x_0 + \alpha z)$ de α para $\alpha \in \mathbb{R}$ y $z \in D$ alcanza su mínimo en $\alpha = 0$. Por lo tanto

$$\frac{d}{d\alpha} f(x_0 + \alpha z)\Big|_{\alpha=0} = \langle Ax_0 - b, z\rangle = 0$$

Lector(a): ¡Verifíquelo!

De donde

$$Ax_0 = b$$

dado que D es denso en H y $z \in D$ era arbitrario.

En esta forma se ha establecido la equivalencia entre los dos problemas siguientes: Resolver la ecuación $Ax = b$ y determinar el mínimo de f. En los problemas 3 y 5, D y H están dados respectivamente por:

$$D = \{\phi \in C^2[0,1] \mid \phi(0) = \phi(1) = 0\}; \quad H = L^2[0,1] \quad y$$

$$D = \{\phi \in C^2(\Omega); \phi \Big|_{\partial\Omega} = 0\}; \quad H = L^2(\Omega)$$

En estos problemas es posible que no exista $x \in D$ tal que $Ax = b$ para $b \in H$ y por el resultado anterior la funcional correspondiente no alcanzará su mínimo en D; sin embargo, como el operador A es definido positivo se puede extender a un operador autoadjunto, por ejemplo la extensión de Friedrichs, (véanse referencias [3] y [4]) de tal manera que f alcance su mínimo. El dominio extendido es un espacio de Hilbert y será denotado por $H^{1,2}(\Omega)$, (véanse [5] y [8]).

Nota. Sea Ω como antes y $C_0^\infty(\Omega)$ denota el espacio vectorial de las funciones f que tienen derivadas de todos los órdenes, las cuales se anulan fuera de un compacto K (el cual depende de f) en Ω.

En $C_0^\infty(\Omega)$ definamos el producto interior

$$\langle \phi, \psi \rangle = \int_\Omega \left(\phi\psi + \frac{\partial\phi}{\partial x_1}\frac{\partial\psi}{\partial x_1} \quad \frac{\partial\phi}{\partial x_2}\frac{\partial\psi}{\partial x_2} \right) dx_1 dx_2$$

Ahora consideremos la norma inducida por este producto interior, a saber

$$\|\phi\| = \langle \phi, \phi \rangle^{1,2}$$

Definiremos al espacio $H_0^{1,2}(\Omega)$ como la completación

de $\{C_0^\infty(\Omega), |\phi|\}$.

PROBLEMA 6.

Consideremos el siguiente problema con valores en la frontera en ecuaciones diferenciales ordinarias

$$A(\phi) = (a_2(t)\phi'')'' - (a_1(t)\phi')' + a_0(t)\phi = b(t)$$

$$\phi(a) = 0, \qquad \phi(b) = 0 \tag{7}$$

$$\phi''(a) = 0, \qquad \phi''(b) = 0$$

donde

$$a_0, a_1, a_1', a_2, a_2', a_2'' \in C[a,b], \quad a_0(t) > 0.$$

$$a_1(t) \geq 0, \quad a_2(t) \geq \gamma > 0 \quad \text{en } [a,b]$$

y

$$b(t) \in L^2[a,b].$$

Algunas observaciones

1. $D = \{\phi \in C^4[a,b] \mid \phi \text{ satisface (7)}\}$ es un subespacio denso de $L^2[a,b]$.
2. El operador $A: D \to L^2[a,b]$ es definido positivo.

La simetría del operador A es una consecuencia de la

siguiente igualdad:

$$<A\phi,\psi> = \int_a^b (a_2\phi''\psi'' + a_1\phi'\psi' + a_0\phi\psi)dt$$

La desigualdad de definido positivo, se basa en la desigualdad de Poincaré (véase [5]).

3. Como se mencionó, existe una extensión autoadjunta de A de manera que la ecuación $A(t) = b(t)$ tiene solución generalizada para todo $b \in L^2|a,b|$. Para obtener dicha solución generalizada se minimiza la funcional

$$f(\phi) = \int_a^b \left[\frac{a_2(t)\phi''(t)^2}{2} + \frac{a_1(t)\phi(t)'^2}{2} + \frac{a_0(t)\phi(t)^2}{2} - b\phi(t)\right] dt,$$

la cual resulta de integrar por partes $<A\phi(t),\phi(t)>$ en la siguiente relación

$$f(\phi) = \frac{1}{2} <A\phi,\phi> - <b,\phi>.$$

PROBLEMA DE DIRICHLET PARA LA ECUACION DE LAPLACE

El uso de los métodos variacionales en ecuaciones diferenciales parciales se remonta a Dirichlet con el conocido principio de Dirichlet (véase [6]), el cual se describe a continuación:

Sea Ω como antes en $\mathbb{R}^2$ y sea g una función con-

tinua definida en la frontera $\partial\Omega$ de Ω. El problema de Dirichlet consiste en encontrar una función ϕ continua en $\Omega U \partial\Omega$ tal que

$$\nabla^2\phi = \frac{\partial^2\phi}{\partial x_1^2} + \frac{\partial^2\phi}{\partial x_2^2} = 0 \quad \text{en} \quad \Omega$$

$$\phi(x_1,x_2) = g(x_1,x_2) \quad \text{si} \quad (x_1,x_2) \in \partial\Omega$$

Un método para estudiar el problema de Dirichlet fue dado a conocer por Riemann en 1851 que consistía en lo siguiente:

Sea M la familia de funciones definidas en $\Omega U \partial\Omega$ tal que:

Son continuas en Ω con primera derivada continua en Ω y coinciden con la función g dada en $\partial\Omega$.

Para $\phi \in M$ defina la integral

$$D[\phi,\phi] = \iint_\Omega \left[\left(\frac{\partial\phi}{\partial x_1} \right)^2 + \left(\frac{\partial\phi}{\partial x_2} \right)^2 \right] dx_1 dx_2 \qquad (8)$$

Consideremos la función $\phi_0 \in M$ que tenga la propiedad de hacer que la integral alcance su valor mínimo. Si $\phi \in C^2(\Omega)$ y se anula en la frontera, entonces $(\phi_0 + t\phi) \in M$ y además $D[\phi_0+t\phi,\phi_0+t\phi] \geq D[\phi_0,\phi_0]$ que desarrollando se obtiene $2tD[\phi_0,\phi] + t^2D[\phi,\phi] \geq 0 \Rightarrow D[\phi_0,\phi] = 0$. De otra manera podríamos escoger t con magnitud y signo apropiado para que

se cumpla la desigualdad contraria.

Usando la siguiente identidad de Green (véase [1])

$$\iint_{\Omega}\left(\frac{\partial\phi}{\partial x_1}\frac{\partial\phi_0}{\partial x_1}\quad\frac{\partial\phi}{\partial x_2}\frac{\partial\phi_0}{\partial x_2}\right)dx_1dx_2 = \int_{\partial\Omega}\phi\,\frac{\partial\phi_0}{\partial n}\,ds - \iint_{\Omega}\phi\nabla^2\phi_0 dx_1dx_2$$

se concluye que

$$\iint_{\Omega}\phi\nabla^2\phi_0 dx_1dx_2 = 0;\ \text{para todo}\ \ \phi\in C^2(\Omega)\ \ \text{tal que}\ \ \phi\Big|_{\partial\Omega} = 0$$

lo que es posible solo si $\nabla^2\phi_0 = 0$ en $\mathring{\Omega}$.

Por lo tanto ϕ_0 es solución del problema de Dirichlet.

A la suposición de que el mínimo de (8) existe, Riemann le dió el nombre de "principio de Dirichlet" y lo justificaba argumentando razones físicas. En 1870 Weierstrass (1815-1879) hizo la crítica del método de Riemann, mostrando que es incompleto, puesto que no demuestra la existencia de una función de clase C^2 que minimice la integral (8). El "principio de Dirichlet" tuvo que esperar hasta que Hilbert (1862-1943) lo rehabilitara encontrando restricciones sobre M y $\partial\Omega$ que hacen a M compacto y a $D[\phi,\phi]$ continua, permitiendo entonces que el mínimo de la integral (8) se alcance para alguna función ϕ_0 y podamos en estos casos resolver el problema de Dirichlet. Posteriormente, Ritz propuso un método para aproximarse al mínimo de una funcional, el cual vino a fortalecer los métodos variacionales. Desde entonces los métodos variacionales han experimentaod un gran progreso.

A continuación describiremos brevemente el método de Ritz.

METODO DE RITZ

Ilustraremos el método de Ritz, considerando casos como los ejemplos 3 y 5, para esto supondremos que el dominio de A, $D(A) = D(f)$ es el espacio de Hilbert $H_0^{1,2}(\Omega)$ en el cual la ecuación tiene solución. Si la solución $x_0 \notin D$ le llamaremos solución generalizada de la ecuación $Ax = b$. El método de Ritz nos permite obtener aproximaciones a la solución generalizada x_0 de la ecuación $Ax = b$.

Definición 1. $\{x_n\} \subset H_0^{1,2}(\Omega)$ es una sucesión minimizante para la funcional f si $\lim_{n \to \infty} f(x_n) = \min f(x)$.

Supondremos el siguiente (véase [4])

Resultado 2. Si $\{x_n\}$ es una sucesión minimizante para f entonces $\{x_n\}$ converge a x_0 en $H_0^{1,2}(\Omega)$.

El resultado 2 indica que para resolver la ecuación $Ax = b$ es suficiente construir una sucesión minimizante para f y cualquiera de sus términos puede considerarse como una aproximación a la solución generalizada de la ecuación $Ax = b$. Especialmente el método de Ritz nos permite construir sucesiones minimizantes para la funcional f.

Sea $\{\phi_n\}$ una sucesión en $H_0^{1,2}(\Omega)$ (para muchas situaciones prácticas es suficiente considerar $\{\phi_n\} \subset D$) tal que:

i) $\{\phi_n\}$ es linealmente independiente.

ii) Completa, es decir, para cada $x \in H_0^{1,2}(\Omega)$ y $\varepsilon > 0$ existen $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ tales que

$$\left\| x - \sum_{k=1}^{n} \alpha_k \phi_k \right\|_{H_0^{1,2}(\Omega)} < \varepsilon$$

Para cada n, entero positivo, $H_n = [\phi_1 \ldots \phi_n]$ denota el espacio lineal generado por los primeros n elementos de la sucesión.

El método consiste en construir el mínimo (el cual existe) de f sobre H_n para cada n y en esta forma se obtiene una sucesión que es minimizante (véanse [4] y [5]) para f.

La determinación del mínimo en cada subespacio $H_n \subset D$ nos da lugar a los siguientes sistemas de ecuaciones algebraicos llamados sistemas de Ritz.

$$\sum_{k=1}^{n} \alpha_k \langle A\phi_k, \phi_i \rangle = \langle b, \phi_i \rangle \quad i = 1 \ldots n \tag{9}$$

Nota. En la descripción del método de Ritz no se dice como construir las funciones ϕ_i, pero desde un punto de vista práctico es deseable seleccionar H_n de manera que las ϕ_i sean tales que $\langle A\phi_k, \phi_i \rangle$ sea "fácil" de calcular, que la matriz del sistema (9) sea rala y que dicho sistema esté bien condicionado. Esto es precisamente uno de los aspectos que trata el método del elemento finito (véase [7]).

PROBLEMA 7.

Consideremos el siguiente problema con valores en la frontera

$$A(\phi) = -\frac{d^2\phi}{dt^2} + \phi(t) = -t$$

$$\phi(0) = 0$$

$$\phi(1) = 0$$

En este caso $D = \{\phi \in C^2[0,1] \mid \phi(0) = \phi(1) = 0\}$ y la funcional asociada al problema está dada por:

$$f(\phi) = \int_0^1 \left[\frac{(\phi')^2(t)}{2} \quad \frac{\phi^2(t)}{2} + t\phi(t) \right] dt$$

Sea $\{\phi_n\} \subset D$ definida por $\phi_n(t) = t^n(1-t)$; $n = 1,2,\ldots$ y construyamos el mínimo de f sobre $H_1 = [t(1-t)]$. Substituyendo $x_1(t) = c_1 t(1-t)$ en la funcional se obtiene una función de c_1, a saber $\frac{11}{30} c_1^2 + \frac{1}{6} c_1$ que al minimizarla nos da $c_1 = -\frac{5}{22}$.

Por lo tanto el mínimo de f sobre H_1, se alcanza en:

$$x_1(t) = -\frac{5}{22} t(1-t).$$

Por otro lado, la solución del problema original está dada por:

$$x_0(t) = \frac{e}{e^2-1} (e^t - e^{-t}) - t.$$

Calculando x_0 y x_1 por ejemplo en $t = \frac{1}{2}$, encontramos que

$$x_0(1/2) = -0.057$$
$$x_1(1/2) = -0.056.$$

BIBLIOGRAFIA

[1] Lang, S.; Analysis I, Addison-Wesley, Publishing Co., Reading Mass. 1968.

[2] Mikhlin, S.G.; An Advanced Course of Matheamtical Physics, North-Holland, Pub. Co. Amsterdam, 1970.

[3] Yosida, K.; Funcitonal Analysis, Springer-Verlag, Berlin, 1966.

[4] Martínez, H.; Notas sobre el método de Ritz, C.I.M.A.S., UNAM, México 1974.

[5] Mikhlin, S.G.; The problem of the minimum of a Quadratic Functional Holden-Day, Inc., San Francisco, 1965.

[6] Courant, R.; Dirichlet's Principle, Conformal Mapping and Minimal Surfaces, Nueva York, 1950.

[7] Hernández, D.B.; Análisis Numérico: Aproximación y Algoritmica, Notas del 3er. Coloquio del Departamento de Matemáticas del CIEA-IPN, México, D.F., 1983.

[8] Hahn, G.S.; Distribuciones y Transformada de Fourier, Notas del 3er. Coloquio del Departamento de Matemáticas del CIEA-IPN, México, D.F., 1983.

ESTIMACION DE ESTADOS EN UN REACTOR*

Erick Gamas C.+

Resumen

En el presente trabajo se introduce una alternativa para obtener los estados de un reactor químico (Concentración y Temperatura) operando en régimen transitorio, a partir de mediciones de la temperatura y cálculo de la derivada de la temperatura con el tiempo. Mediante esto se "mide" la tasa de generación de calor, la cual es un término que introduce nolinealidad en las ecuaciones que describen el comportamiento del sistema.

Se analiza el esquema global del problema planteando la solución de las ecuaciones diferenciales del sistema, en función de una proyección de los estados (C y T) usando el método de Colocación Ortogonal (1,2) y resolviendo numéricamente las ecuaciones discretizadas que resultan.

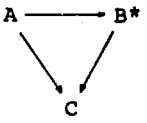
---

* Este reporte es un resumen del trabajo iniciado por Klaus F. Jensen y Jesús Alvarez en la Universidad de Minnesota (USA) y continuado hasta su terminación por el mismo Jesús Alvarez con la colaboración de Erick D. Gamas en el Area de Ingeniería Química de la Universidad Autónoma Metropolitana, Unidad Iztapalapa. El artículo completo está citado en la referencia 9.

+ Universidad Autónoma Metropolitana, Unidad Iztapalapa.

1. EL PROBLEMA

Supongamos que queremos producir una cierta sustancia B, la cual se puede obtener mediante la siguiente reacción química,



*Producto requerido

Esta reacción se lleva a cabo en un reactor tubular que opera en flujo pistón (3,4), de tal forma que los estados del reactor son sólo función de la posición axial (reactor unidimensional),

$$C = C(z)$$
$$T = T(z).$$

Se desea diseñar un esquema de control del reactor (Figura 1) en base al conocimiento de la concentración de salida de este ($C_B(z = L)$). Esto hace necesario predecir la evolución temporal de dichos estados con objeto de calibrar un estimador dinámico de los estados del reactor.

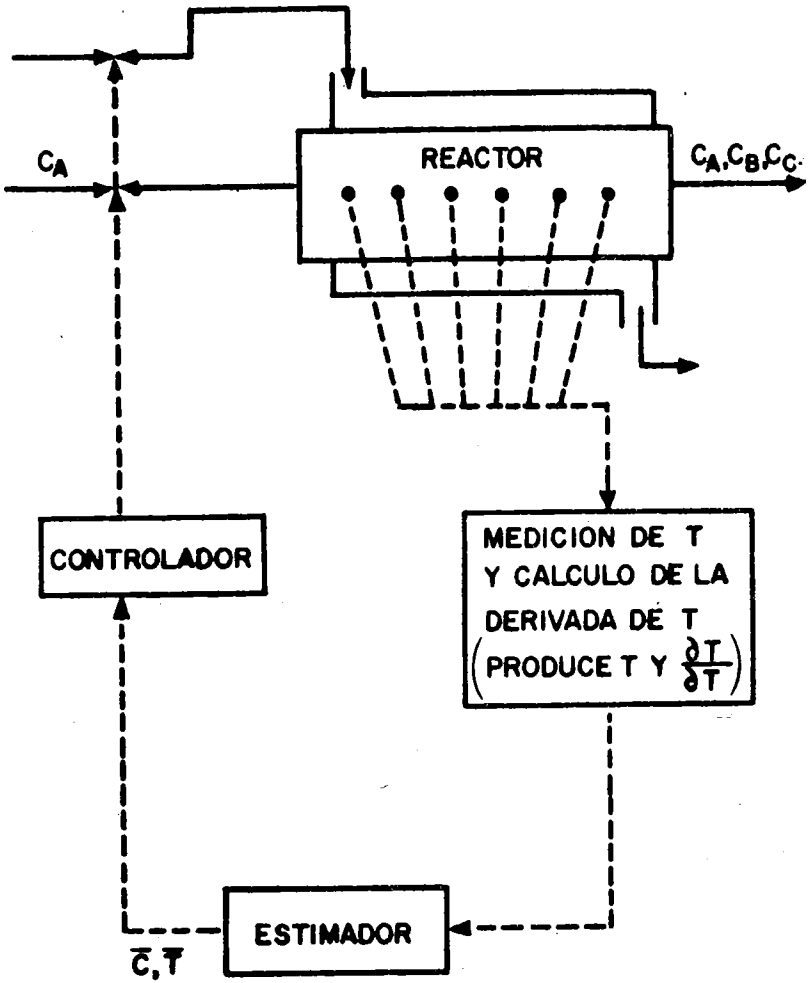


Figura 1. Esquema de Control

Para llevar a cabo la estimación de la concentración de B a la salida del reactor es necesario disponer de un modelo del sistema, el cual está representado por las ecuaciones que gobiernan el comportamiento de las sustancias durante su paso por el reactor. Estas ecuaciones se obtienen considerando las leyes de Fick y Fourier (para la difusión de masa y calor, respectivamente) (5,6,7); las ecuaciones están representadas en forma adimensional por el siguiente sistema de ecuaciones diferenciales parciales:

Balance de Materia

$$\frac{\partial \xi}{\partial t} = \frac{\partial^2 \xi}{\partial s^2} - P_{em} \frac{\partial \xi}{\partial s} - \alpha \mathbb{R}(\xi,\eta) \qquad 0 < s < 1 \quad t > 0 \qquad (1)$$

Balance de Energía

$$\frac{\partial \eta}{\partial t} = \frac{1}{L_e} \frac{\partial^2 \eta}{\partial s^2} - P_{eh} \frac{\partial \eta}{\partial s} + \alpha\beta \mathbb{R}(\xi,\eta) + \gamma(\eta_w - \eta) \qquad 0 < s < 1 \quad t > 0, \qquad (2)$$

donde, P_{em}, P_{eh}, α, β, L_e son constantes adimensionales dadas por:

$$P_{em} = \frac{LV}{D} \qquad \alpha = \frac{L^2 \mathbb{R}(C_r, T_r)}{DC_r} \qquad L_e = \frac{P_{em}}{P_{eh}}$$

$$P_{eh} = \frac{LV\rho C_p}{K_L} \qquad \beta = \frac{(-\Delta H)C_r}{\rho C_p T_r}$$

Las ecuaciones están sujetas a las siguientes condiciones:

objetivo se desarrolla la temperatura en función de $m+2$ términos (incluyendo las dos condiciones de frontera) utilizando una expansión por polinomios de interpolación de Lagrange ($P_i(s)$) (8),

$$T(s,t) \approx \sum_{i=0}^{m+1} P_i(s_i) T_i(t) \tag{3}$$

$$\mathbb{R}(s,t) \approx \sum_{i=0}^{m} P_i(s_i) \mathbb{R}_i(t) \tag{4}$$

donde $T_i(t)$ son las mediciones y $\mathbb{R}_i(t)$ es lo que se quiere conocer.

Sustituyendo (3) y (4) en (2) (los puntos $s_0, s_1, \ldots, s_{m+1}$ se eligen de tal forma que el residuo resultante se anule en estos puntos, los cuales son los ceros del $m+2$-ésimo miembro de la familia de polinomios usada) se obtiene la siguiente ecuación

$$\dot{T}_i(t) = \frac{1}{L_e} \sum_{j=0}^{m+1} P_j^{(2)}(s_i) T_i(t) - P_{eh} \sum_{i=0}^{m+1} P_j^{(1)}(s_i) T_j(t) +$$

$$+ \alpha\beta \mathbb{R}(s_i,t) + \gamma(T_w - T_i(t)) \tag{5}$$

Dado que $i = 1, \ldots, m$, se obtienen m ecuaciones (una por cada medición); el sistema de ecuaciones resultante puede ser representado en forma matricial de la siguiente forma:

$$1)\quad \begin{aligned} &\frac{\partial \xi}{\partial s} = 0 \qquad s = 1 \\ &\qquad\qquad\quad t > 0 \\ &\frac{\partial \eta}{\partial s} = 0 \end{aligned}$$

$$2)\quad \begin{aligned} &\frac{\partial \xi}{\partial s} = P_{em}(\xi - \xi_i) \qquad s = 1 \\ &\qquad\qquad\qquad\quad t > 0 \\ &\frac{\partial \eta}{\partial s} = P_{eh}(\eta - \eta_i) \end{aligned}$$

$$3)\quad \begin{aligned} &\xi(s,0) = \xi_0 \qquad 0 < s < 1 \\ &\qquad\qquad\quad t = 0 \\ &\eta(s,0) = \eta_0 \end{aligned}$$

2. LA SOLUCION

La estructura de la solución que se propone es la siguiente:

a) Estimación de la nolinealidad.

b) Simulación del reactor (en sustitución del sistema físico).

c) Estimación de los estados ($\bar{C}$ y $\bar{T}$).

a) Estimación de la nolinealidad.- Observando las ecuaciones que describen el comportamiento del sistema (1) y (2), en ellas aparece un término en común, la tasa de reacción

$\mathbb{R}(\xi,\eta)$; este término describe la rapidez con que se produce o se consume una sustancia (ya sea producto o reactivo) y las expresiones que lo describen son variadas, según el comportamiento químico de la sustancia, por ejemplo (3,4):

$$\mathbb{R} = kC_A^n \qquad \text{(Modelo Ley de Potencia)}$$

$$\mathbb{R} = \frac{kC_B}{1+kC_B} \qquad \text{(Modelo Langmuir-Heinshelwood)}$$

La característica principal en estas expresiones que describen a $\mathbb{R}$ es que aparece un término en común, una constante $k = k(T)$ relacionada por la expresión de Arrhenius (4),

$$k = A \exp\left(-\frac{E_a}{RT}\right)$$

Este tipo de funcionalidad exponencial en k, y por tanto en $\mathbb{R}$, provocan que las ecuaciones (1) y (2) sean ecuaciones diferenciales parciales no lineales.

Debido a la nolinealidad de las ecuaciones, se propone la estimación de $\mathbb{R}$ en base a mediciones de temperatura y cálculo de su derivada con respecto al tiempo usando estas mediciones.

La estimación de la nolinealidad se basa en la medición de m valores de la temperatura, y la evaluación de $(\partial T/\partial t)$ usando estas mediciones, a lo largo del reactor; con este

$$\dot{T}(t) = \Gamma T(t) + \alpha\beta \mathbb{R}(t) + w.$$

De esta ecuación se puede despejar $\mathbb{R}(t)$ para obtener la ecuación siguiente:

$$\mathbb{R}(t) = \frac{1}{\alpha\beta} \left| \dot{T}(t) - \Gamma T(t) - w \right| \qquad (6)$$

donde,

Γ - contiene las derivadas de los polinomios de Lagrange

α, β, w - son parámetros del sistema

Dado que los valores conocidos son $\dot{T}(t)$ y $T(t)$, el valor de $\mathbb{R}(t)$ se estima según la ecuación (6).

b) Simulación del reactor.- La simulación del sistema que se está estudiando es un recurso útil en este tipo de estudios, dado que la implementación, puesta en marcha y operación del sistema físico, es un problema independiente del objetivo de este trabajo.

La simulación se lleva a cabo resolviendo las ecuaciones diferenciales que describen el comportamiento del sistema cuando éste se encuentra en estado estacionario (es decir, cuando $\partial T/\partial t = 0$); esto se hace con el propósito de obtener los perfiles de concentración y temperatura que representan

el estado estacionario, el cual va a ser alterado aumentando bruscamente la concentración (un pulso) para llevar a cabo la etapa de estimación.

Las ecuaciones que resultan al igualar las derivadas con respecto al tiempo a cero son:

Balance de Materia

$$\frac{\partial^2 \xi}{\partial s^2} - P_{em} \frac{\partial \xi}{\partial s} - \alpha \mathbb{R}(\xi,\eta) = 0 \qquad (7)$$

$$0 < s < 1$$

Balance de Energía

$$\frac{1}{L_e} \frac{\partial^2 \eta}{\partial s^2} - P_{eh} \frac{\partial \eta}{\partial s} + \alpha\beta \mathbb{R}(\xi,\eta) + \gamma(\eta w - \eta) = 0 \qquad (8)$$

$$0 < s < 1$$

Estas ecuaciones están sujetas a las siguientes condiciones de frontera:

1)
$$\frac{\partial \xi}{\partial s} = 0 \qquad s = 1$$
$$\frac{\partial \eta}{\partial s} = 0 \qquad t > 0$$

2)
$$\frac{\partial \xi}{\partial s} = P_{em}(\xi - \xi_i) \qquad s = 0$$
$$\frac{\partial \eta}{\partial s} = P_{eh}(\eta - \eta_i) \qquad t > 0$$

Es decir, se obtiene un sistema de ecuaciones diferenciales ordinarias no lineales.

La forma en que se plantea la solución de las ecuaciones es la siguiente:

Proyección de los Estados.- Los estados del reactor se proyectan usando el método de colocación ortogonal (con lo cual se reduce el orden de las ecuaciones) de tal forma que el sistema de ecuaciones diferenciales ordinarias se aproxima por un sistema de ecuaciones algebraicas no lineales. Las expresiones para la proyección son las siguientes (1,2,8):

$$\xi(s,t) \sim \sum_{j=0}^{n+1} \ell_j(s)\xi_j(t) \qquad (9)$$

$$\eta(s,t) \sim \sum_{j=0}^{n+1} \ell_j(s)\eta_j(t) \qquad (10)$$

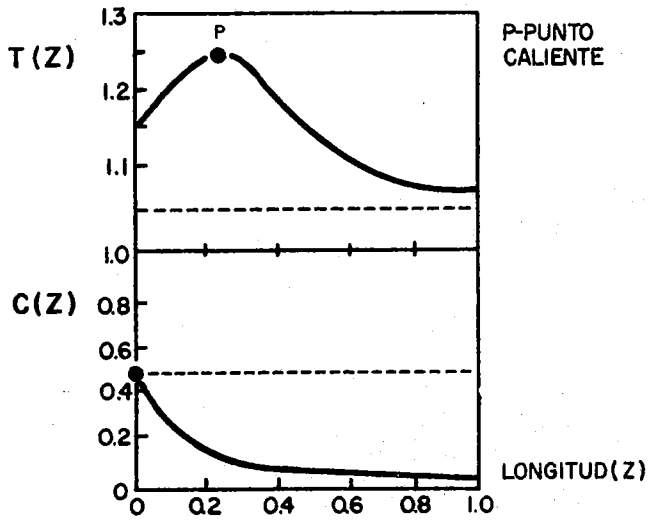
ξ_i, η_i - representan la concentración y temperatura en el i-ésimo punto de colocación

$\ell_i(s)$ - son los polinomios de interpolación de Lagrange

Si se sustituyen (9) y (10) en (7) y (8) y se obliga a que los residuos obtenidos se anulen en $s_0, s_1, \ldots, s_m, s_{m+1}$ (los nodos de colocación), se obtiene un sistema de ecuaciones algebraicas no lineales, el cual se resuelve usando el método de Newton-Raphson.

El resultado que da la aplicación del método de Newton-Raphson son los perfiles de concentración y temperatura en estado estacionario, es decir, se obtienen $C(z)$ y $T(z)$ (o bien, $\xi(z)$ y $\eta(z)$, en forma adimensional).

La gráfica de los perfiles en estado estacionario está representada en la Figura 2, (9).



c) Estimación de estados.- El objetivo de la estimación de estados es obtener valores de $C(z,t)$ y $T(z,t)$, a los cuales se denotará como $\bar{C}(z,t)$ y $\bar{T}(z,t)$, que sean aproximados a los valores "reales" de $C(z,t)$ y $T(z,t)$. Esto se hace por medio de la manipulación adecuada de las ecuaciones (1) y (2) utilizando la estimación obtenida de la nolinealidad, una proyección usando colocación ortogonal y aplicando la teoría de estimadores (10).

Usando colocación ortogonal para proyectar las funciones de estado del reactor y sustituyendo estas expresiones (ecuaciones (9) y (10)) en las ecuaciones (1) y (2), el resultado es que las ecuaciones diferenciales parciales se aproximan por ecuaciones diferenciales ordinarias no lineales.

En las ecuaciones obtenidas se sustituye el término de estimación de la nolinealidad (ecuación (6)) para obtener el siguiente sistema de $2n$ ecuaciones diferenciales ordinarias no lineales (n puntos interiores de colocación):

$$\frac{d\underline{c}}{dt} = \underline{G}_C\underline{C} + \underline{h}_C - \alpha\mathbb{R}^*(t) \qquad *\text{Estimada}$$

$$\frac{d\underline{T}}{dt} = \underline{g}_T\underline{T} + \underline{h}_T + \alpha\beta\mathbb{R}^*(t) \qquad \underline{c}, I, \underline{h}_C, \underline{h}_T, \mathbb{R}^* \quad \mathbb{R}^n$$

$$\underline{G}_C, \underline{g}_T \quad \mathbb{R}^{n\times n}$$

$\underline{G}_C, \underline{G}_T$ - Matrices constantes que representan la parte lineal

del operador (incluyen difusión y convección)

$\underline{h}_C, \underline{h}_T$ - Vectores constantes que contienen las condiciones de entrada y en la pared en el reactor.

Este sistema de ecuaciones diferenciales ordinarias no lineales se resuelve utilizando como datos de entrada los valores del estado estacionario, los cuales son alterados mediante un pulso en la concentración para observar el comportamiento del estimador. Los perfiles estimados $\bar{C}$ y $\bar{T}$ se obtienen resolviendo el sistema de EDO's aplicando el método de Runge-Kutta de 4ª orden (se obtiene $\bar{C}(z,t)$ y $\bar{T}(z,t)$).

3. RESULTADOS

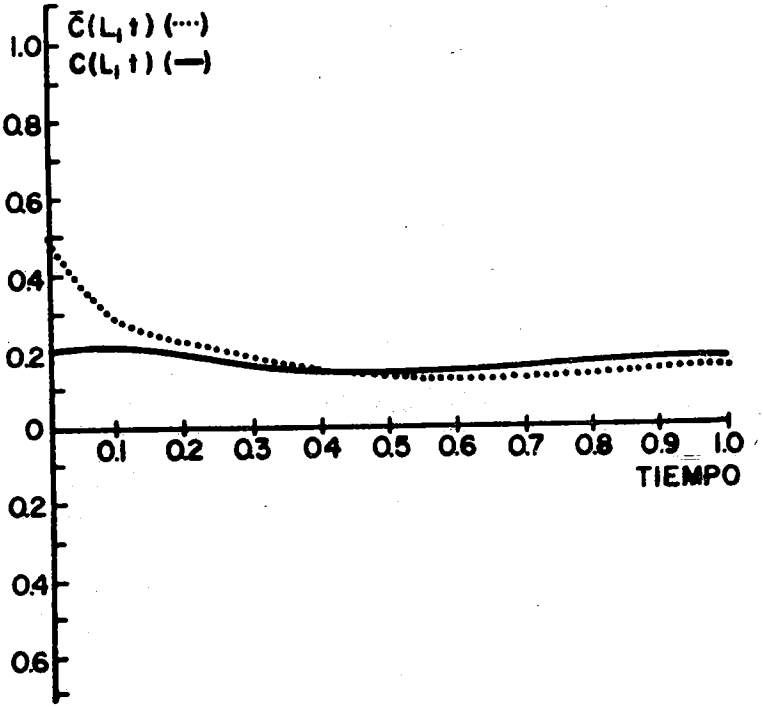
El comportamiento del estimador se evaluó comparando los valores de concentración estimados en $z = L$ (por ser el objetivo de control) con los valores obtenidos de la integración de las ecuaciones (1) y (2) después de haber sido proyectadas mediante el método de colocación ortogonal, sin incorporar la estimación de la nolinealidad, usando el método de Runge-Kutta. Los resultados de ambas integraciones se muestran en la Gráfica 1.

Dado que cuando se hacen mediciones, se hacen cálculos o se modela un sistema de cometen errores, se probó la sensibilidad del estimador a desviaciones inducidas en el mode-

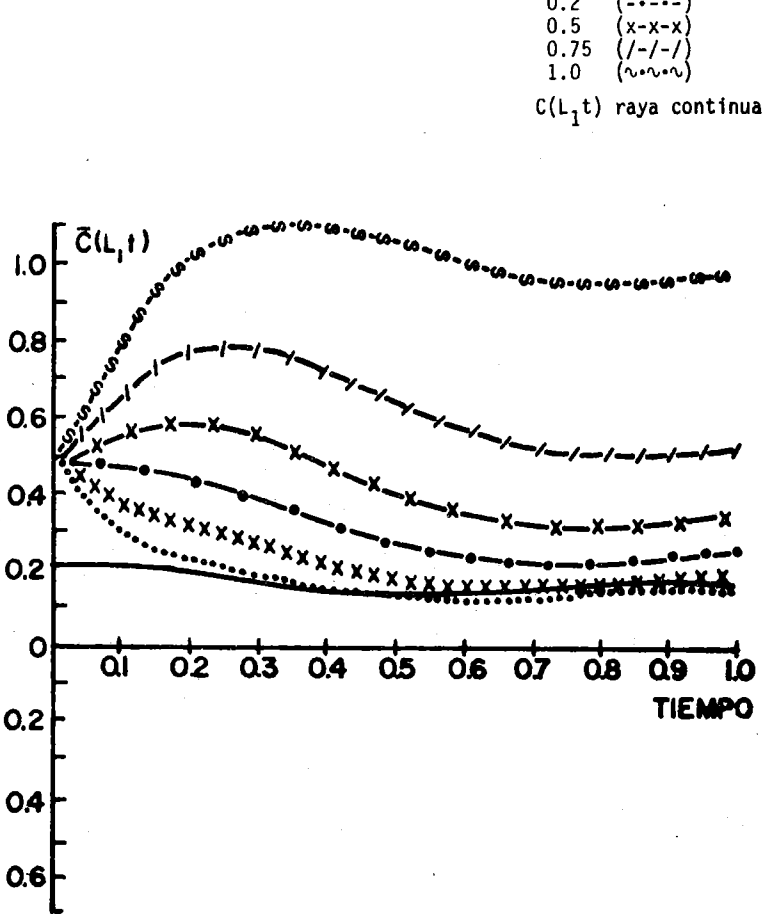
lo, en las mediciones de la temperatura y en el cálculo de la derivada de ésta, observándose que tanto el modelo como las mediciones presentan gran sensibilidad a las desviaciones, mientras que las mediciones de la derivada son completamente insensibles a éstas; esto fué predicho "a priori" analizando las ecuaciones del estimador (9). Con esto se concluye que es necesario diseñar una técnica de medición que permita que estas presenten la menor desviación posible del valor real. Estos resultados se muestran en las gráficas 2, 3 y 4.

4. CONCLUSIONES

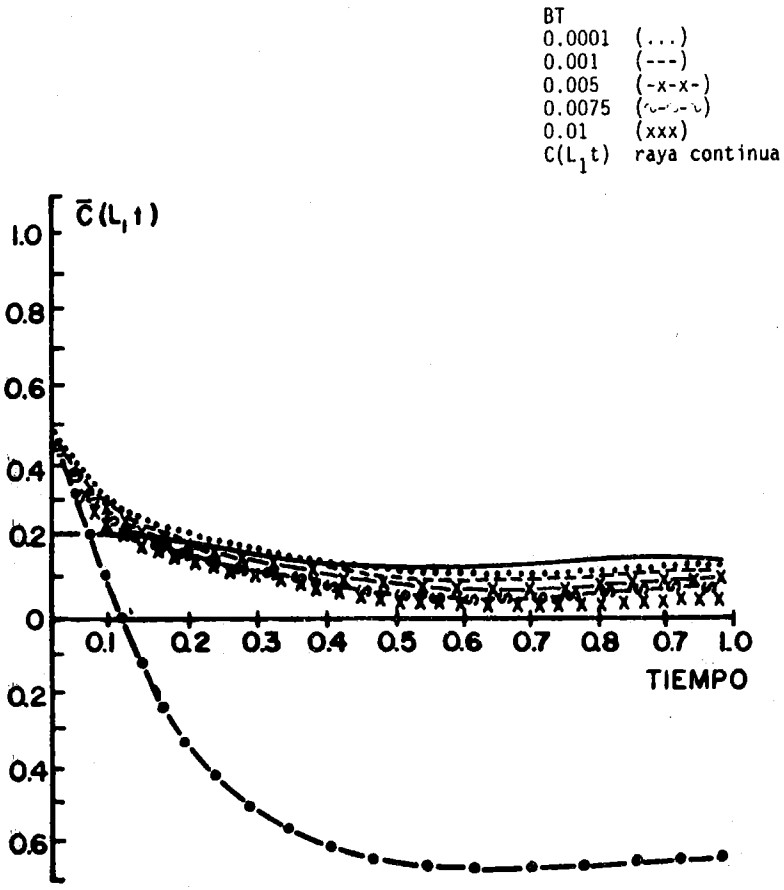
En este trabajo se muestra la utilidad que presenta el tratamiento matemático de los problemas físicos, ya que se logra obtener una gran información acerca del comportamiento del sistema sin que sea necesaria la construcción del mismo (el reactor para nuestro caso específico); así mismo, se puede conocer información del comportamiento del sistema que no puede ser obtenida experimentalmente.



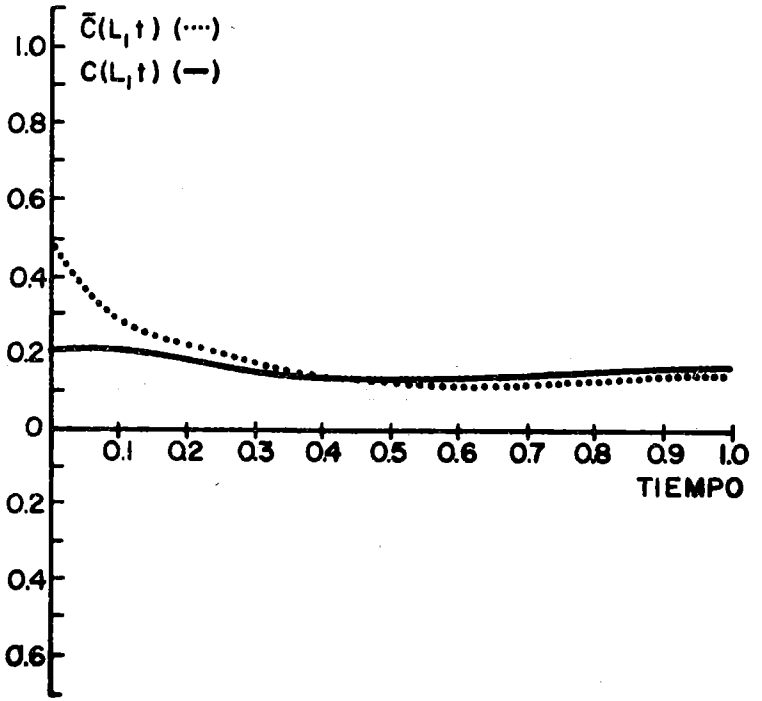
Gráfica 1. Comportamiento del estimador (línea punteada) con respecto a los valores "reales".



Gráfica 2. Desviación en el Modelo BM (BT = BTD = 0)



Gráfica 3. Desviaciones en las mediciones BT (BM = BTD = 0).



Gráfica 4. Desviación en la derivada de la temperatura BTD (BM = BT = 0).

BIBLIOGRAFIA

1. Villadsen, J.V. y Stewart, W.E.; "Chemical Engineering Science, V. 22, pp. 1483-1501 (1967).
2. Finlayson, B.A.; "Orthogonal Collocation in Chemical Reaction Engineering", Catalysis Review Science and Engineering, 10(1), 69-138 (1974).
3. Hougen and Watson; "Chemical Process Principles", Kinetics and Catalysis, part three, Ed. Wiley (1959).
4. Carberry, J.J.; "Chemical and Catalytic Reaction Engineering", McGraw Hill (1976).
5. Ozisik, M.N., "Transferencia de Calor", McGraw Hill 1979.
6. Sherwood, T.K., Pigford, R.L., Wilke, C.R.; "Mass Transfer", McGraw Hill/Kogakusha, 1975.
7. Bird, R.B., Stewart, W.E., Lightfoot, E.N.; "Transport Phenomena", Wiley, 1960.
8. Villadsen, J. y Michelsen, M.; "Solution of Differential Equation-Models by Polynomial Approximation", Prentice Hall, New Jersey, 1978.
9. Jensen, K.F., Alvarez, J., Gamas, E.D.; "Estimation of Reactor States, a Combined Interpolation and Filtering Approach", aceptado para publicación en Chemical Engineering Science, 1983.
10. Gelb, A., Kasper, J.F., Nash, R.A., Price, C.F., Sutherland, A.A.; "Applied Optimal Estimation", The M.I.T. Press, Cambridge 1978.

Notación

A	Factor de Frecuencia
BM	Desviación en el modelo
BT	Desviación en las mediciones
BTD	Desviación en la derivada
C	Concentración dimensional
$\bar{C}$	Concentración estimada
Cp	Capacidad calorífica
D	Difusividad
Ea	Energía de activación
ΔH	Entalpía de reacción
k	Constante cinética de reacción
k_L	Difusividad térmica
L	Longitud del reactor
Le	Grupo adimensional de Lewis
m	Número de mediciones
n	Número de puntos de colocación
Pi(s)	Polinomios de Lagrange
P_{em}, P_{eh}	Grupo adimensional de Péclet para masa y calor, respectivamente
R	Tasa de reacción
R	Constante de los gases
s	Longitud adimensional
T	Temperatura dimensional
$\bar{T}$	Temperatura dimensional estimada
t	Tiempo
V	Volumen del reactor

α, β, γ	Parámetros del reactor que surgen al adimensionalizar las ecuaciones
ξ	Concentración adimensional
ρ	Densidad del fluido
η	Temperatura adimensional
η_w	Temperatura adimensional en la pared del reactor

LA ESTRUCTURA DE UN LIQUIDO; SOLUCION NUMERICA DE ECUACIONES INTEGRALES NO-LINEALES

L. Mier y Terán C.*

1. INTRODUCCION [1]

Uno de los propósitos de la física de los líquidos consiste en determinar su estructura microscópica cuando éstos se encuentran en equilibrio termodinámico. Debido a la agitación térmica, el significado del término estructura, en un líquido no corresponde, como en el caso del sólido, a un arreglo rígido de las moléculas que lo componen sino que más bien tiene un sentido probabilístico. En un líquido las posiciones de las moléculas no están tan definidas como en una red cristalina pero tampoco se encuentran distribuidas en forma totalmente azarosa. En la determinación, tanto teórica como experimental, de la estructura de un líquido es de gran importancia el conocimiento de la función de distribución radial, g(r). Dicha función es proporcional a la probabilidad de que, tomando como centro a una molécula cualquiera del fluido, encontremos a otra a una distancia r, medida desde el centro de la primera. Naturalmente la función g(r) depende drásticamente de la temperatura, la densidad, y desde luego, de las características particulares de las mo-

* Departamento de Física, Universidad Autónoma Metropolitana, Unidad Iztapalapa, Apdo. Postal 55-534, 09340 México, D.F.

léculas que componen el sistema.

La mecánica estadística proporciona varios caminos para determinar en forma aproximada la función de distribución radial. Una de esas rutas, la de las aproximaciones integrales, requiere resolver ecuaciones integrales no-lineales en forma numérica, pues sólo se han encontrado soluciones analíticas de ellas en casos muy particulares. Con el propósito de calcular la función g(r) según estas aproximaciones integrales, se han aplicado varios métodos numéricos iterativos que han proporcionado una cantidad enorme de información, sin embargo esos métodos presentan serias dificultades de convergencia en algunas regiones termodinámicas de interés. Por ejemplo, en la región de densidades altas y temperaturas bajas, es decir, justamente en el estado líquido.

En este trabajo presentamos un método numérico iterativo que está basado en técnicas de elemento finito y que ha resultado ser muy eficiente en la solución de esa clase de ecuaciones integrales.[2] Asimismo mostraremos algunos resultados obtenidos con ese método numérico para una aproximación conocida como Aproximación Esférica Promedio (MSA).[3] El modelo potencial intermolecular utilizado es el de "pozo cuadrado" que definiremos más adelante.

2. LA ESTRUCTURA DEL FLUIDO Y SUS PROPIEDADES TERMODINAMICAS

El potencial de interacción entre dos partículas de un fluido simple, es decir, aquel que está formado por molécu-

las con simetría esférica, puede dividirse en dos regiones importantes; una parte repulsiva que opera a distancias cortas y otra atractiva que se manifiesta a distancias intermedias. En un fluido simple el potencial decae rápidamente a cero para distancias grandes. En la fig. 1.a) se presenta una forma típica de un potencial de esa clase. Correspondientemente, en la fig. 1.b) se muestra la forma cualitativa de la g(r) de un líquido denso. Debido a la fuerte repulsión intermolecular que se presenta a distancias cortas, la g(r) tiende a valer cero cuando r tiende a cero. El primer máximo de g(r) se encuentra localizado en una posición cercana al mínimo del potencial. Los "picos" de la función g(r) son los remanentes, en el estado líquido, de las capas de vecinos que rodean a una molécula en una posición de una red cristalina. A distancias grandes la función g(r) tiende al valor unidad, lo cual significa que, a esas separaciones intermoleculares, la correlación entre partículas se vuelve nula. Muy pocos fluidos reales caen dentro de la clasificación de fluidos simples. Sólo las moléculas de los gases nobles presentan muy aproximadamente esa forma de potencial intermolecular.

Como ya se mencionó, en este trabajo se utilizó al potencial de pozo cuadrado como modelo de la interacción entre pares de moléculas. Ese modelo está definido como

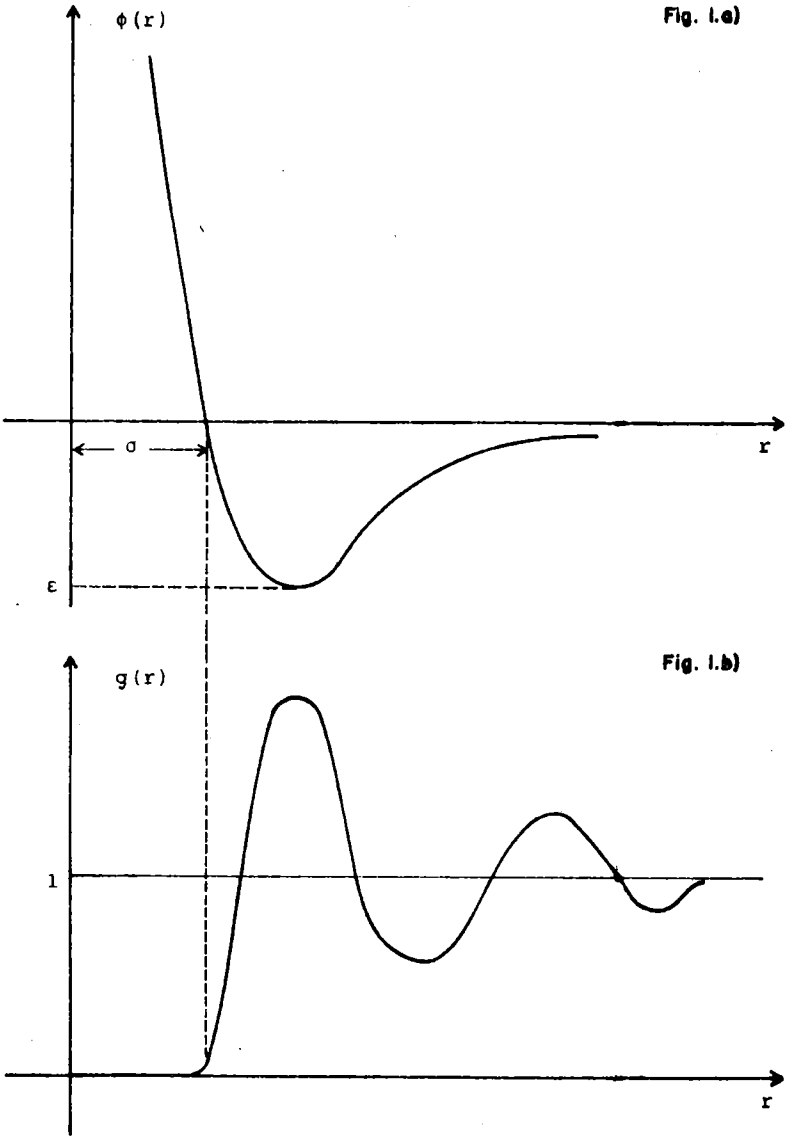


Figura 1

$$\phi(r) = \begin{cases} \infty, & r < \sigma, \\ -\varepsilon, & \sigma < r < \lambda\sigma, \\ 0, & r > \lambda\sigma. \end{cases} \tag{1}$$

A pesar de que a primera vista el modelo de interacción definido por la ec. (1) parece muy poco realista, se trata de una función de tres parámetros $(\sigma,\varepsilon,\lambda)$ que ha demostrado tener las características físicas necesarias para adecuarse a las moléculas de los fluidos simples. De este modo, mediante el potencial de pozo cuadrado se han podido representar las propiedades de fluidos simples como el Argón o el Kripton en forma bastante precisa.

En la mecánica estadística de los líquidos se hace generalmente la suposición de que la energía configuracional de un sistema de N moléculas se puede escribir como

$$\Phi = \sum_{i<j}^{N}\sum^{N} \phi(r_{ij}). \tag{2}$$

Esta es la suposición conocida como de "aditividad por pares". Si se utiliza esa suposición, el conocimiento de la función g(r) permite calcular las propiedades termodinámicas del sistema. Por ejemplo, la energía interna, U, se puede obtener como

$$\frac{U}{NkT} = \frac{3}{2} + \frac{2\pi\rho}{kT}\int_0^\infty \phi(r)g(r)r^2dr \tag{3}$$

en donde T es la temperatura absoluta, k es la constante de Boltzmann y $\rho = N/V$ es la densidad de partículas encerradas en un volumen V. A su vez, la ecuación de estado para la presión, pV, del sistema se puede escribir como

$$\frac{pV}{NkT} = 1 - \frac{2\pi}{3kT}\rho\int_0^\infty r^3\frac{d\phi(r)}{dr}g(r)dr. \tag{4}$$

La compresibilidad isotérmica, $K_T \equiv \frac{1}{\rho}\left(\frac{\partial\rho}{\partial p}\right)_T$, es una cantidad termodinámica de importancia, que está relacionada con la función g(r) a través de la ecuación:

$$4\pi\rho\int_0^\infty (g(r)-1)r^2dr = \rho kTK_T - 1. \tag{5}$$

A la función g(r)-1 se le suele denotar por h(r) y es conocida como la función total de correlación. Es importante hacer notar que, a diferencia de las ecs. (3 y 4) el potencial $\phi(r)$ no aparece explícitamente en la ec. (5), esto se debe a que en la deducción de esta expresión no es necesario utilizar la suposición de aditividad por pares.

De este modo, la función g(r) no sólo nos da información acerca de la estructura del fluido sino también sobre su termodinámica. Más aún, la función g(r) puede ser obtenida mediante experimentos de dispersión de rayos X o de neutrones. Esto último permite, en principio, la comparación de la g(r) calculada mediante alguna teoría con la obtenida en forma experimental.

Interesados en el tratamiento de los fenómenos que ocurren en la región crítica de un fluido, Ornstein y Zernike definieron en 1914 a la función directa de correlación, c(r). Debido a que la correlación total, h(r) = g(r)-1, entre pares de partículas del fluido es generalmente de mayor alcance que el potencial, lo cual es matemáticamente inconveniente, Ornstein y Zernike buscaron una función que tuviera un alcance semejante al de $\phi(r)$. Definieron así a la función directa de correlación entre dos partículas, 1 y 2, por medio de la relación integral[1]

$$h(r_{12}) = c(r_{12}) + \rho \int_V c(r_{13})h(r_{23})d\vec{r}_3. \tag{6}$$

La integral en el miembro derecho de la ec. (6) se efectúa sobre el volumen total del sistema, V, y $d\vec{r}_3$ representa al elemento de volumen en que se encuentra una tercera partícula cualquiera del fluido. El sentido físico de esta definición se ve más claramente cuando se reemplaza sucesivamente a la misma ec. (6) dentro de su propio integrado. Resulta entonces que

$$h(r_{12}) = c(r_{12}) + \rho \int_V c(r_{13})c(r_{23})d\vec{r}_3 + \rho^2 \int_V\int c(r_{13})c(r_{34})c(r_{42})d\vec{r}_3 d\vec{r}_4 + \ldots$$

De esa manera podemos ver que la correlación total, $h(r_{12})$,

se descompone en una correlación directa entre 1 y 2 y una correlación indirecta a través de todas las posibles cadenas de correlaciones directas.

La relación de Ornstein-Zernike, ec. (6), es, como ya dijimos, una definición de la función c(r) y no puede ser resuelta si no se cuenta con una relación adicional entre c(r) y g(r). Varios autores han sugerido diversas formas aproximadas para esta relación de "cerradura". Entre las más importantes se encuentran la de "cadena hipertejida",

$$(7) \qquad \text{(HNC)} \qquad c(r) = g(r) - 1 - g(r) - \frac{\phi(r)}{kT} ;$$

la de Percus-Yevick,

$$(8) \qquad \text{(PY)} \qquad c(r) = g(r) \left(1 - \exp \frac{\phi(r)}{kT} \right) ,$$

y la de la aproximación esférica promedio

$$(9) \qquad \text{(MSA)} \qquad \begin{aligned} g(r) &= 0, \qquad r < \sigma, \\ c(r) &= - \frac{\phi(r)}{kT}, \qquad r > \sigma. \end{aligned}$$

La aproximación esférica promedio está definida para potenciales de coraza repulsiva impenetrable, es decir, para aquellos en los que

$$\phi(r) = \infty, \quad r < \sigma. \tag{10}$$

El potencial de pozo cuadrado, definido en la ec. (1), tiene esa forma.

La combinación de relaciones de "cerradura", como las de las ecs. (7), (8) y (9) con la relación de Ornstein-Zernike, ec. (5), produce ecuaciones integrales no lineales para la función g(r). En el caso MSA, si se aprovecha la simetría esférica del potencial $\phi(r)$ para llevar a cabo las integrales sobre las coordinadas angulares, se obtiene que[3]

$$-r = C(r) + 2\pi\rho \left[\int_0^{\sigma} C(s) \int_{|r-s|}^{r+s} H(t)\,dt\,ds - \beta \int_{\sigma}^{\infty} s\phi(s) \int_{s-r}^{r+s} H(t)\,dt\,ds \right], \quad r < \sigma, \tag{11}$$

$$H(r) = -r\beta\phi(r) + 2\pi\rho \left[\int_0^{\sigma} C(s) \int_{|r-s|}^{r+s} H(t)\,dt\,ds - \beta \int_{\sigma}^{\infty} s\phi(s) \int_{|r-s|}^{r+s} H(t)\,dt\,ds \right], \tag{12}$$

para $r > \sigma$. En las dos ecuaciones anteriores se ha usado que, $C(r) = r\,c(r)$ y $H(r) = r\,h(r)$ y $\beta = 1/kT$. En la siguiente sección describiremos un método numérico que es aplicable a la solución de este tipo de ecuaciones integrales.

3. EL METODO NUMERICO

En la solución numérica de las ecuaciones integrales de la teoría de líquidos se han utilizado, con mayor o menor éxito, un cierto número de métodos. Entre los más importantes podemos mencionar el de iteración directa[4] y el de transformación de Fourier[5]. Recientemente[2] ha sido desarrollado otro método numérico que, basado en técnicas de elemento finito, ha demostrado poseer un gran poder de convergencia y versatilidad para diversas aplicaciones. Por ser éste el método utilizado en este trabajo trataremos de describirlo con cierto detalle en esta sección.

El algoritmo que formularemos aquí está deseñado para resolver ecuaciones de la forma

$$(13)\qquad v(y(r)) + \int w(\bar{\alpha};s,y(s))\,(K(|\bar{r}-\bar{s}|;\bar{\beta})\,y(|\bar{r}-\bar{s}|)-1)\,d\bar{s} = 0,$$

en donde $y(r)$ es la función incógnita y $\bar{\alpha}$ y $\bar{\beta}$ son dos conjuntos de parámetros. Integrada sobre las coordenadas angulares la ecuación anterior puede ser reescrita como

$$(14)\qquad v(y(r)) + 2\pi \int_0^{\infty}\int_{|r-s|}^{r+s} w(\bar{\alpha};s,y(s))\cdot[K(t;\bar{\beta})y(t)-1]\,\frac{ts}{r}\,dtds = 0,$$

en donde $t = |\bar{r}-\bar{s}|$. Usando ahora las definiciones

$$Y(r) \equiv ry(r),$$

$$V(r) \equiv rv(r),$$

y

$$W(\bar{\alpha};s,y(s)) \equiv sw(\bar{\alpha};s,y(s)),$$

la ec. (14) se transforma en

$$(15)\quad V(Y(r)) + 2\pi \int_0^{\infty}\int_{|r-s|}^{r+s} W(\bar{\alpha},s,Y(s))\cdot[k(t;\bar{\beta})Y(t)-t]dtds = 0.$$

El propósito del método de elemento finito es el de transformar una sóla ecuación, digamos

$$(16)\qquad L(y(r)) = 0,$$

en un sistema algebraico de ecuaciones. Esto se hace mediante una subdivisión del dominio de las variables del problema en subdominios de tamaños y formas apropiadas y utilizando formas funcionales simples para representar a la solución en cada subdominio o elemento.

En este caso la función incógnita y(r) es aproximada mediante una base de funciones linealmente independientes $\{\psi^j\}$, es decir,

$$(17)\qquad y^a(r) = \sum_{j=1}^{N} q_j\psi^j(r).$$

Los coeficientes $\{q_i\}$, que son inicialmente desconocidos, son los valores de la solución en las intersecciones o nodos

de los elementos.

De acuerdo al criterio de minimización conocido como colocación[6], la función residual, $L(y^a)$, es reducida a cero en los nodos de los elementos

$$\int L(Y^a)\ (r-r_i)d^3r = 0 \tag{18}$$

en donde $\delta(r-r_i)$ es la función delta de Dirac.

Aunque en principio, el dominio de la función y(r) es infinito, podemos imponer una condición asintótica sobre la ecuación integral (15). Esto es, requeriremos a la función y(r) tomar una forma asintótica para valores grandes de r. La posibilidad más simple consiste en suponer que la función se haga cero para r mayor que un cierto valor R. Los efectos de esta condición asintótica sobre la solución, pueden ser estimados variando el valor de R.

Para resolver la ec. (15), es posible dividir el intervalo $0 \leq r \leq R$ en elementos finitos elegidos adecuadamente y usar una base de funciones lineales definidas por

$$\psi^j(r) = \begin{cases} (r-r_{j-1})/(r_j-r_{j-1}), & r_{j-1} \leq r \leq r_j, \\ (r_{j+1}-r)/(r_{j+1}-r_j), & r_j \leq r \leq r_{j+1}, \\ 0 \quad \text{en el resto.} \end{cases}$$

en donde r_j es el valor de r en el nodo j. La localización de los nodos, $\{r_j\}$, puede elegirse de modo de concentrar más

elementos en aquellas porciones del dominio en donde la función varía más fuertemente. Si se desea obtener una aproximación mejor a la solución, es posible buscar posiciones óptimas para los nodos o simplemente incrementarlos en número.

Para formar el sistema de ecuaciones algebraicas $F_i(\ q_j\ ;\bar{\alpha},\bar{\beta}) = 0,\ i = 1,\ldots,N$, para los coeficientes del desarrollo (17), es necesario sustituir a esa aproximación en la ec. (15) y aplicar el criterio de minimización dado en la ec. (18). Este conjunto de ecuaciones no-lineales puede ser resuelto por un método iterativo, como el de Newton, por ejemplo. Como es bien sabido, el método de Newton proporciona un sistema lineal de ecuaciones para los coeficientes del desarrollo en la iteración n+1, en términos de los mismos coeficientes en la iteración n. En notación compacta, esto es

$$\underline{\underline{J}}(\bar{q}^{(n)})\ (\bar{q}^{(n+1)}-\bar{q}^{(n)}) = -\ \bar{F}(\bar{q}^{(n)}),$$

en donde $\underline{\underline{J}}$ es la matriz Jacobiano. Ese proceso iterativo debe continuarse hasta que la norma Euclideana[7], de la diferencia entre dos iteraciones sucesivas se vuelve menor que un número pequeño, Δ, estipulado previamente,

$$\left|\bar{q}^{(n+1)}-\bar{q}^{(n)}\right| = \left|\ \sum_{j=1}^{N}\ (q_j^{(n+1)}-q_j^{(n)})^2\ \right|^{1/2} < \Delta.$$

Este proceso requiere una estimación inicial $\bar{q}^{(0)}$ suficien-

temente precisa para caer en el dominio de convergencia del método. Encontrar una buena aproximación inicial puede ser difícil en algunos casos, pero es frecuente que se tengan soluciones asintóticas para algún valor límite de los parámetros. Una vez que se obtiene la solución para ciertos valores de los parámetros, es posible contar con buenas aproximaciones iniciales para otros valores de los mismos, mediante una técnica de continuación paramétrica. Si, por ejemplo, se desea obtener la solución $\bar{q}(\alpha_0+\Delta\alpha)$ del sistema algebraico en un valor del parámetro α, $\alpha_0+\Delta\alpha$, cuando ya se cuenta con la solución $q(\alpha_0)$ en el valor α_0, podemos resolver el sistema lineal dado por

$$\underline{\underline{J}}(\bar{q}(\alpha_0))\,(\bar{q}(\alpha_0+\Delta\alpha)-\bar{q}(\alpha_0)) = -\bar{G}(q(\alpha_0)),$$

en donde

$$G_i(\bar{q}(\alpha_0)) = \partial F_i/\partial\alpha$$

Esta última derivada se evalúa en $\alpha = \alpha_0$.

4. RESULTADOS[3]

Para ilustrar la velocidad y precisión del método, así como las cualidades predictivas de la ecuación integral resuelta, en esta sección presentaremos algunas soluciones de

la ecuación integral (MSA) para el modelo de interacción de pozo cuadrado (1), con $\lambda = 1.5$. Para obtener esas soluciones, se utilizó la estrategia de progresar isotérmicamente desde soluciones de baja densidad hasta las de muy alta densidad utilizando la técnica de continuación paramétrica descrita anteriormente. Con el criterio de convergencia $\Delta = 10^{-6}$ se obtuvieron soluciones en una a tres iteraciones para saltos en densidad reducida $\rho\sigma^3$ de 0.1. Es importante hacer notar que ese poder de convergencia persistió incluso en las regiones del diagrama de fases tradicionalmente difíciles y aún en la región de inestabilidad termodinámica.

La convergencia lograda con el método de descrito, contrasta grandemente con la que se obtiene mediante el método de iteración directa o de Picard que requiere hasta de varios cientos de iteraciones para lograr el mismo nivel de precisión a altas densidades y bajas temperaturas. Con 151 nodos igualmente espaciados y una distancia de corte $R = 7.5\sigma$. (La función de correlación se supone igual a uno para $r > R$), se obtuvieron soluciones en 21 isotermas de temperaturas entre $\beta\varepsilon = 0.0$ y 2.0. La densidad reducida $\rho\sigma^3$ se varió entre cero y 1.05. Esta última corresponde a un líquido muy denso. Es conveniente analizar el comportamiento de las funciones $h(r)$ y $c(r)$ en varias situaciones de temperatura y densidad. La fig. (2) muestra la función $g(r)$ a tres distintas densidades $\rho\sigma^3 = 0.1$, 0.4 y 0.7 a la temperatura supercrítica $\beta\varepsilon = 0.6$. La de menor densidad, 0.1, muestra una estructura casi

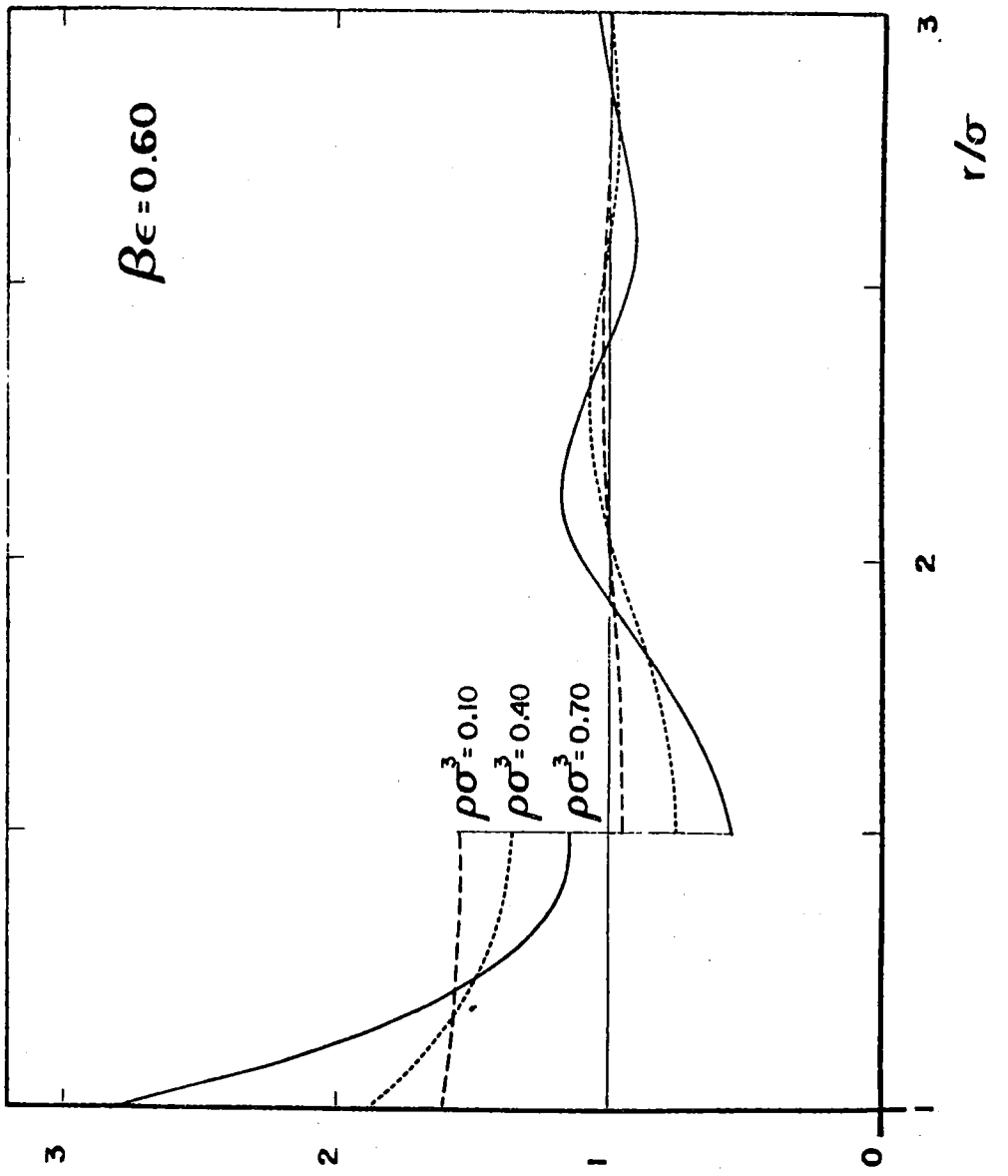

$\beta\epsilon = 0.60$
$\rho\sigma^3 = 0.10$
$\rho\sigma^3 = 0.40$
$\rho\sigma^3 = 0.70$
g(r)
r/σ
0
1
2
3

Figura 2

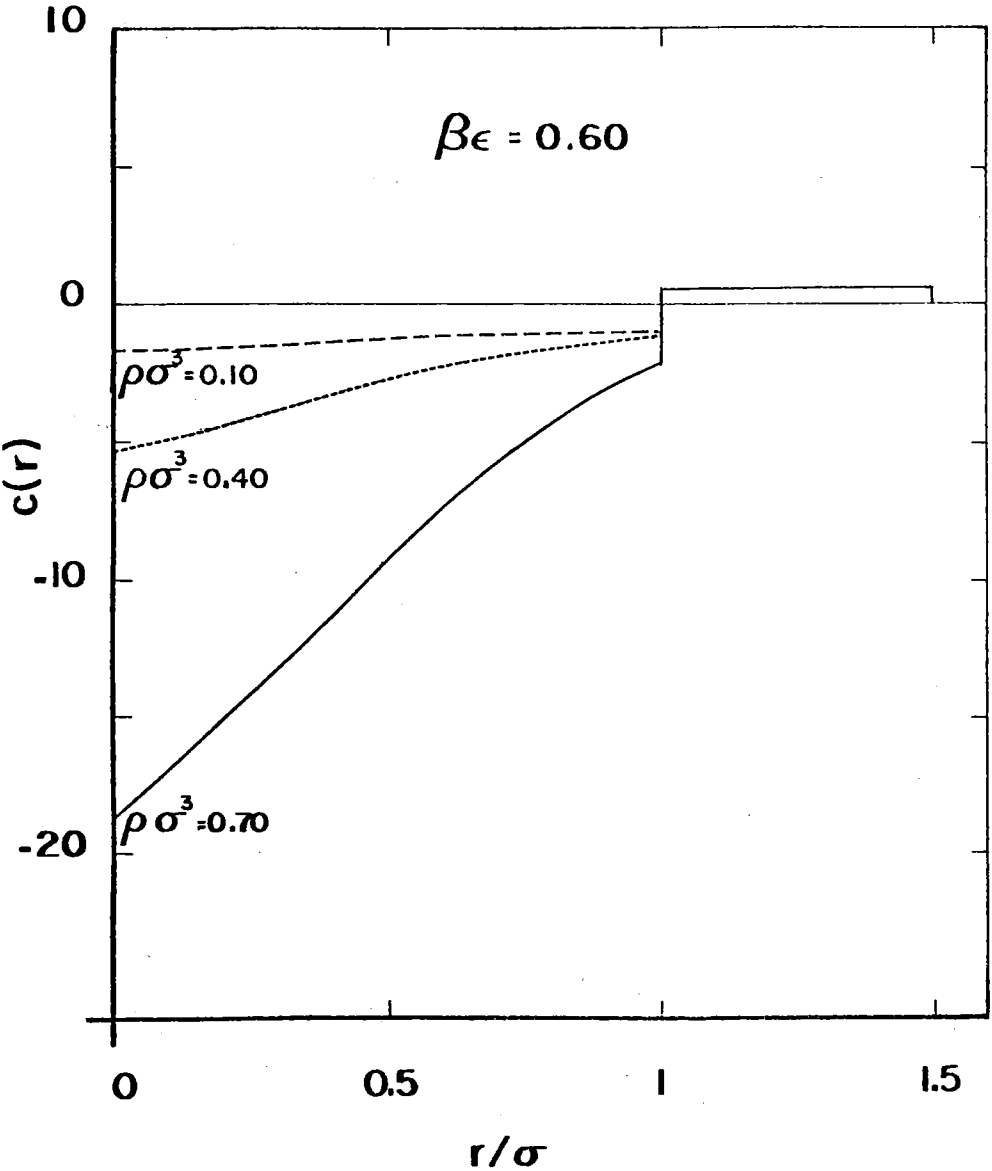

10
0
-10
-20
c(r)
$\beta\epsilon$ = 0.60
$\rho\sigma^3$=0.10
$\rho\sigma^3$=0.40
$\rho\sigma^3$=0.70
0
0.5
1
1.5
r/σ

Figura 3

inexistente. La de densidad intermedia. 0.4, muestra oscilaciones más pronunciadas que se vuelven mucho más notables en la de alta densidad, 0.7. Las tres funciones tienen un comportamiento oscilatorio en el que la posición del primer pico se desplaza a valores de r/σ menores conforme aumenta la densidad. El valor de $g(r)$ en contacto, $g(\sigma^+)$, se incrementa muy notablemente con la densidad. La fig. (3) muestra las funciones $c(r)$ correspondiente a los mismos tres puntos. En este caso es el valor de $-c(0)$ el que se incrementa notablemente con la densidad.

Para mostrar el efecto de la densidad sobre las funciones de correlación en una temperatura subcrítica, en la fig. (4) aparece la $g(r)$ en las mismas tres densidades anteriores, es decir, $\rho\sigma^3 = 0.1$, 0.4 y 0.7 para $\beta\varepsilon = 1.7$. Esta última es una temperatura cercana a la del punto triple del sistema de pozo cuadrado. Nuevamente se presenta una estructura que va haciéndose más y más pronunciada conforme se incrementa la densidad. La función de densidad intermedia, 0.4, corresponde a un punto dentro de la región espinodal del fluido, es decir, la compresibilidad isotérmica, $k_T = \frac{1}{\rho}(\partial\rho/\partial p)_T$, es negativa allí. Ese hecho se manifiesta en el alcance de $g(r)$. La fig. (5) muestra las funciones $c(r)$ correspondientes a los mismos tres puntos de $\beta\varepsilon = 1.7$.

Por lo que se refiere a las propiedades termodinámicas, puede decirse que hay una coincidencia casi total con las calculadas previamente por otros autores[8]. De todas las

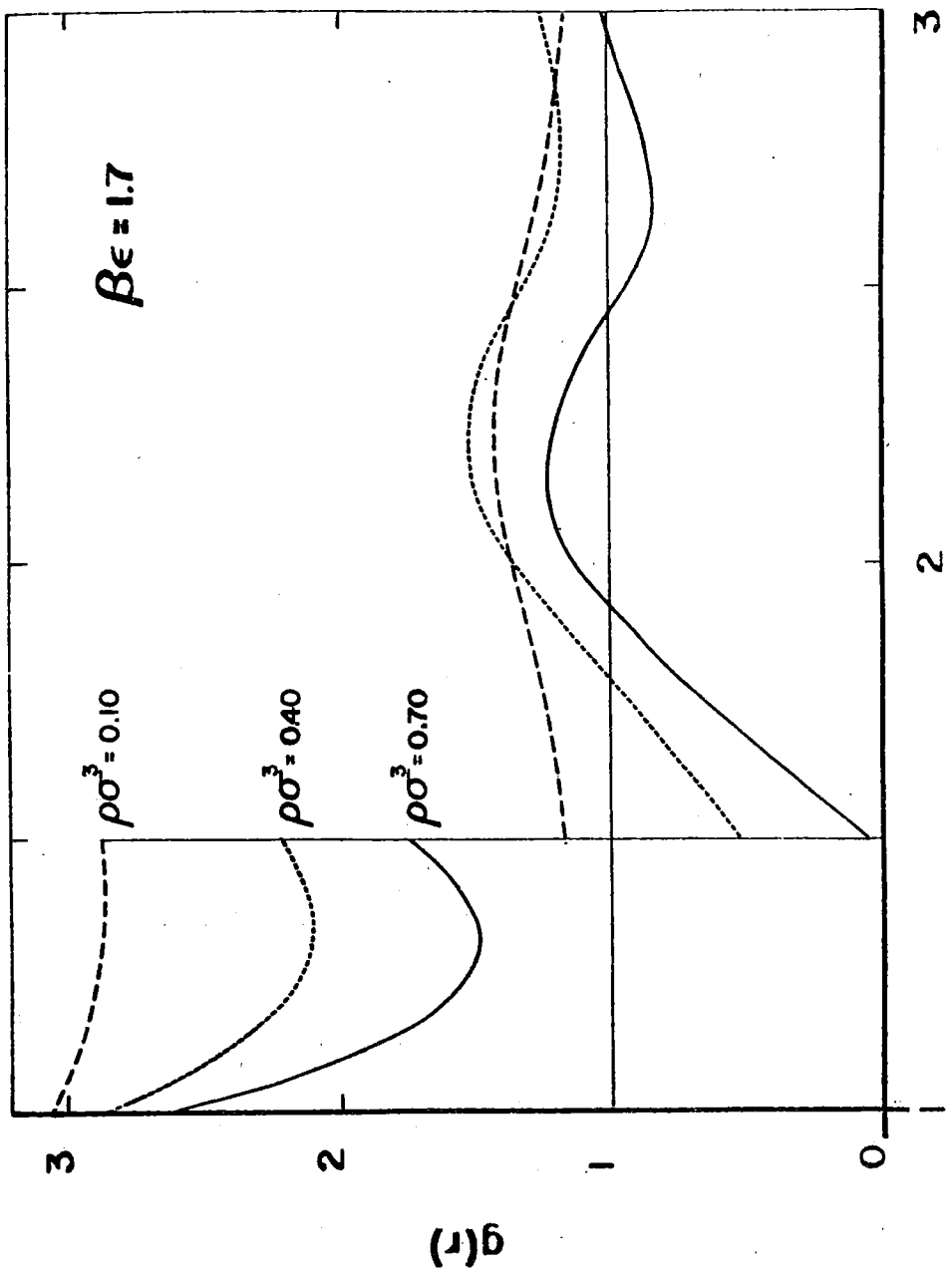


Figura 4

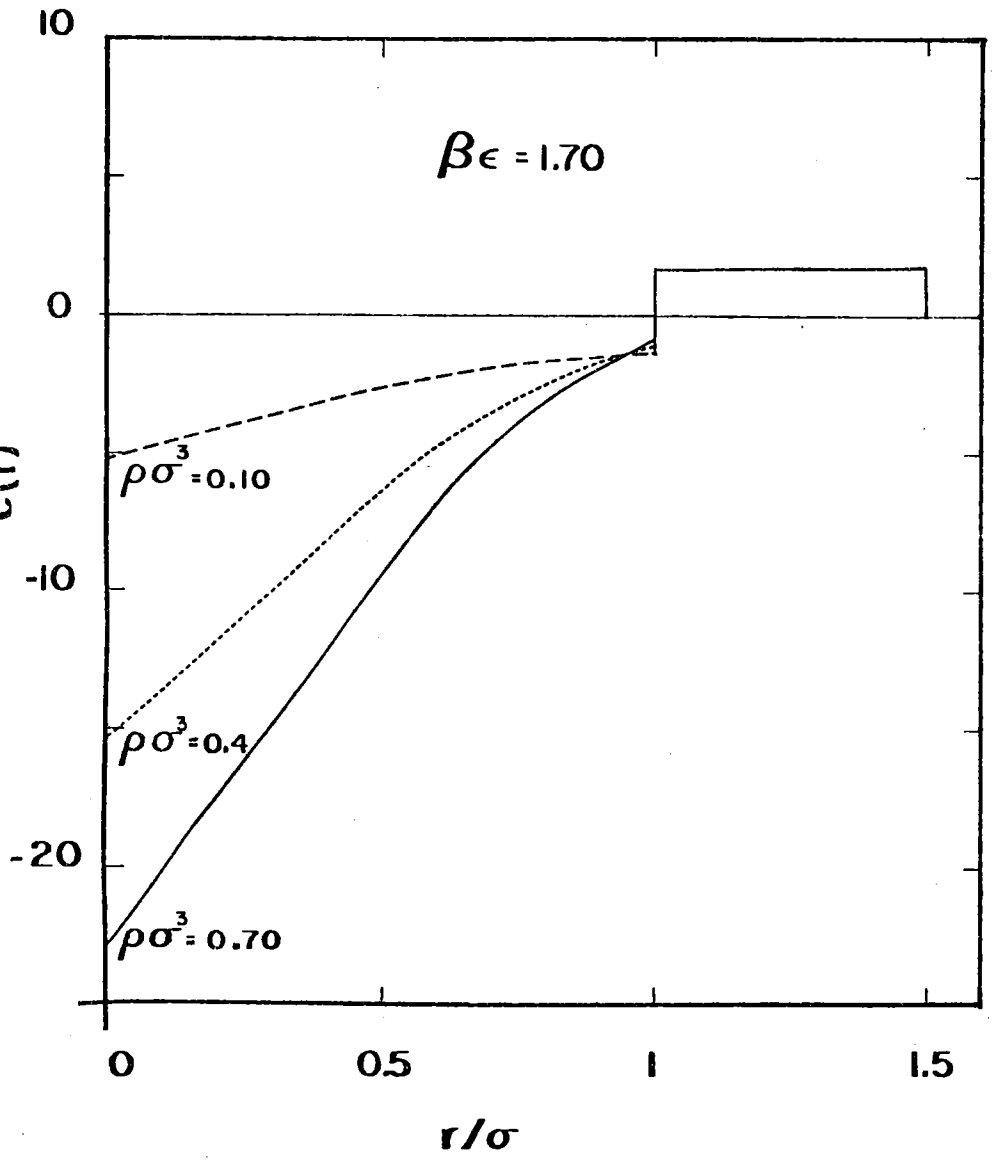


Figura 5

cantidades termodinámicas, la compresibilidad isotérmica es la más difícil de calcular con precisión. Sin embargo, la coincidencia entre los cálculos hechos por Smith et.al.[8] y los calculados con el presente método es excelente. En la región de alta densidad y baja temperatura, $\beta\varepsilon = 2.0$ y $\rho\sigma^3 = 0.9$, por ejemplo, los datos obtenidos en este trabajo discrepan de los de Smith et.al. por menos del 0.3%.

La fig. (6) muestra la presión virial obtenida para seis isotermas, tres de ellas subcríticas. En la Fig. (7) se muestra la cantidad $\beta(\partial p/\partial\rho)_T$ para varias isotermas. En ella se puede ver como $\beta(\partial p/\partial\rho)_T$ es positiva, para todas las densidades, en las isotermas supercríticas. Conforme se disminuye la temperatura, las isotermas llegan a tocar el valor cero en los puntos espinodales, éstos corresponden a los máximos y mínimos de las isotermas de presión mostradas en la fig. (8). La temperatura crítica es aquella en la que $\beta(\partial p/\partial\rho)_T$ se vuelve cero una sóla vez.

Para una ecuación de estado del tipo van der Waals, como la que resulta en nuestro caso, es posible emplear la construcción de Maxwell para determinar la región del equilibrio entre fases. En la fig. (8) se han trazado en líneas punteadas parte de las ramas vapor y líquido de la región de coexistencia.

La región del diagrama termodinámico cercana al punto crítico es motivo de particular interés desde el punto de vista físico. Utilizando el método presentado aquí se ha po-

dido hacer un análisis muy detallado del comportamiento predicho por la aproximación MSA para la compresibilidad isotérmica en esa región[9], resultando ser un comportamiento "clásico" es decir, similar al de la ecuación de estado de van der Waals.

CONCLUSIONES

La relación de Ornstein-Zernike (1) con "cerraduras" como las HNC (7), PY (8) y MSA (9) producen ecuaciones integrales no-lineales. La única forma de evaluar la naturaleza de estas aproximaciones integrales es entender, como con cualquier ecuación no-lineal, la naturaleza de su espacio de soluciones; esto es, el espacio de funciones aumentado por los parámetros, condiciones a la frontera o asintóticas y otras constricciones.

El método utilizado, basado en elemento finito y las técnicas de residuos ponderados, resulta ser muy poderoso, generando soluciones aún en regiones tradicionalmente difíciles. Más aún, con poco esfuerzo computacional adicional, esta técnica puede proporcionar información acerca de la estabilidad y sensibilidad matemática de la ecuación dependiendo de los valores de los parámetros, así como de las posibles bifurcaciones[2] que aparezcan en su espacio de soluciones.

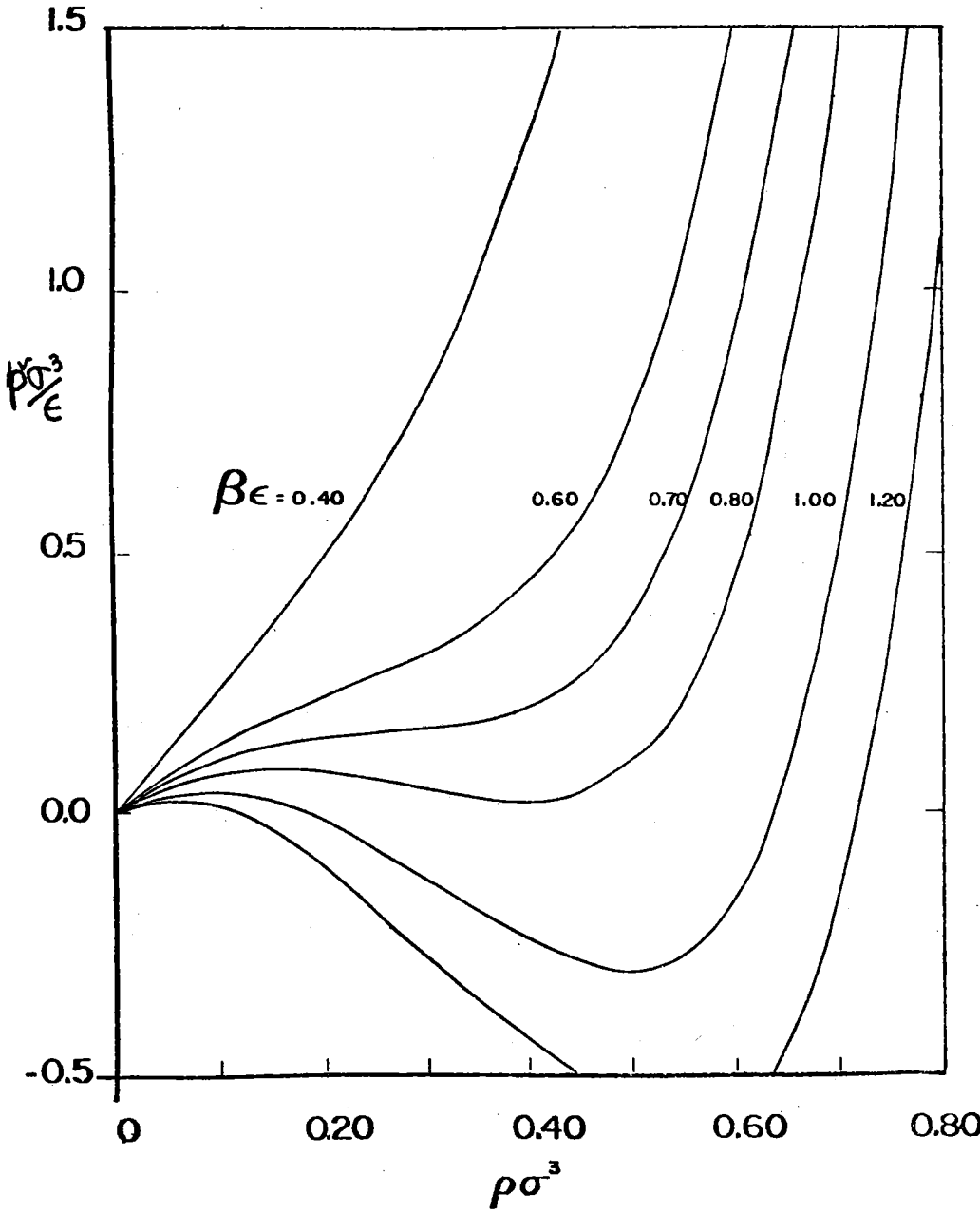

1.5
1.0
0.5
0.0
-0.5
$P\sigma^3/\epsilon$
$\beta\epsilon$ = 0.40
0.60
0.70
0.80
1.00
1.20
0
0.20
0.40
0.60
0.80
$\rho\sigma^3$

Figura 6

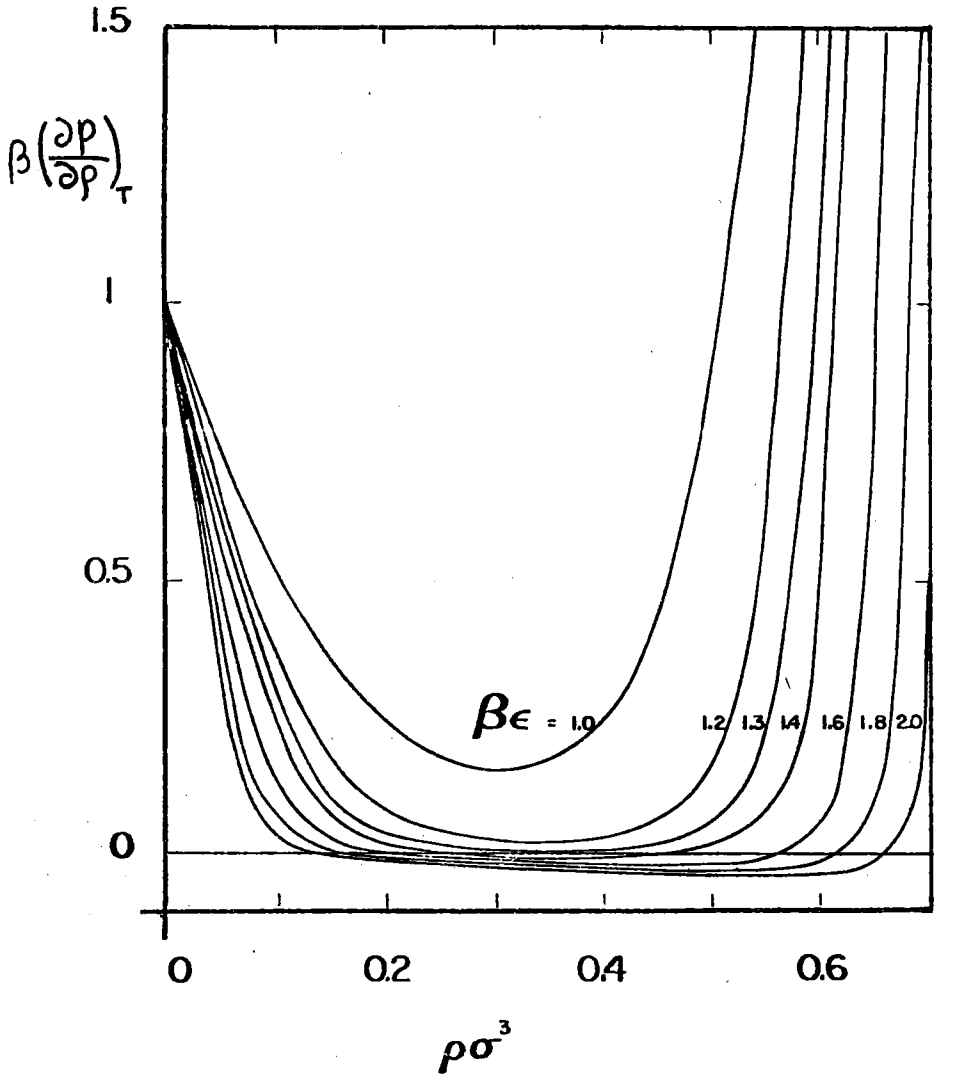

1.5
$\beta\left(\frac{\partial p}{\partial \rho}\right)_T$
1
0.5
0
$\beta\epsilon$ = 1.0
1.2
1.3
1.4
1.6
1.8
2.0
0
0.2
0.4
0.6
$\rho\sigma^3$

Figura 7

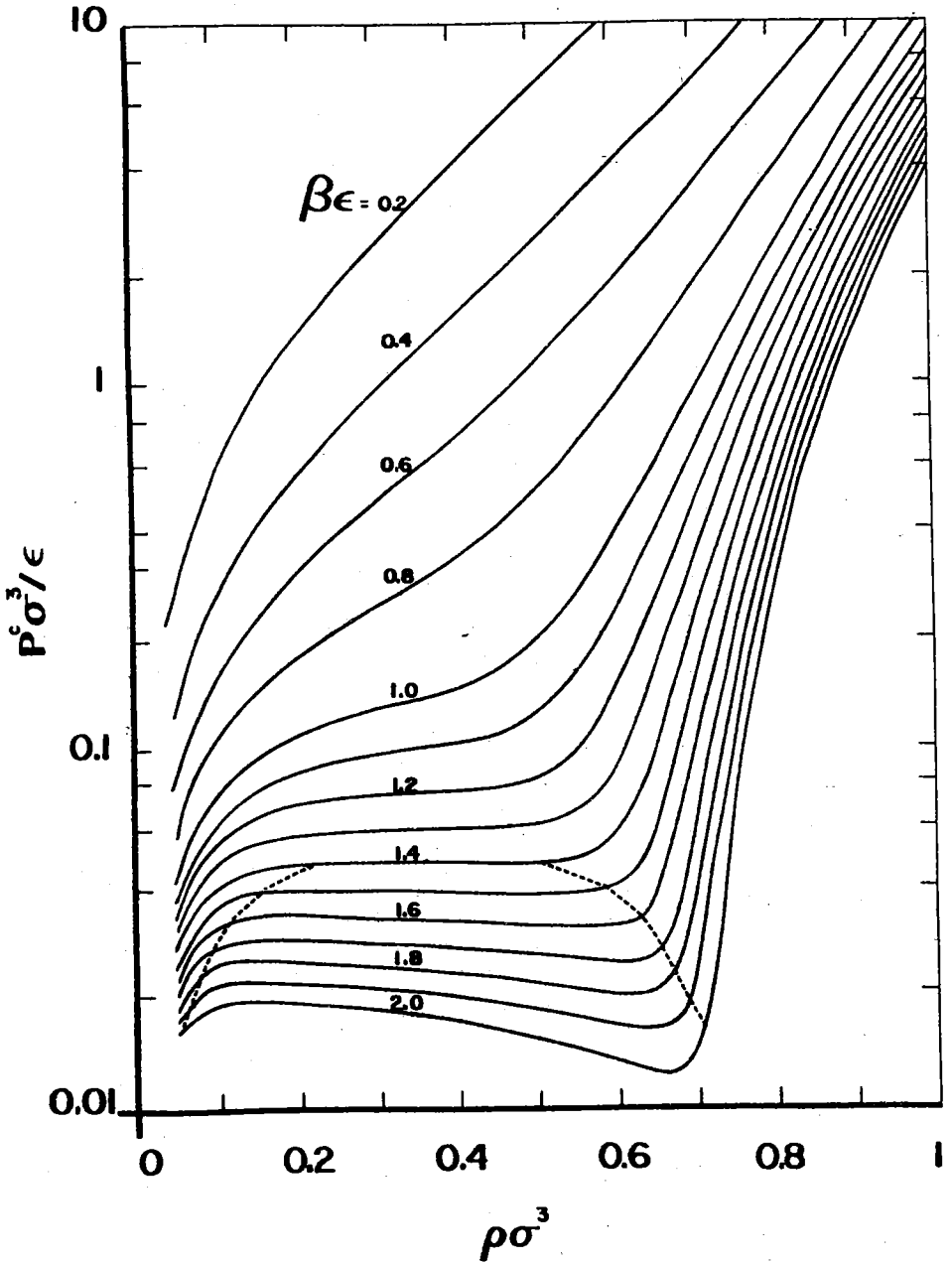

10
1
0.1
0.01
0
0.2
0.4
0.6
0.8
1
$\beta\epsilon$ = 0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
$P^c\sigma^3/\epsilon$
$\rho\sigma^3$

Figura 8

BIBLIOGRAFIA

1. T.M. Reed y K.E. Gubbins, Applied Statistical Mechanics, Chemical Engineering Series, McGraw Hill, New York, 1973.
2. L. Mier y Terán, A.H. Falls, L.E. Scriven y H.T. Davis, Proceedings of the Eighth Symposium on Thermophysical Properties, J.V. Sengers, ed. The American Society of Mechanical Engineers, Vol. I, p. 45 (1982).
3. S.E. Quiñones, Tesis de Licenciatura, Facultad de Ciencias Químicas, Universidad de Guadalajara, Guadalajara, Jal. (1983).
4. A.A. Broyles, J. Chem. Phys., 33, 456(1960); J. Chem. Phys., 35, 493(1961).
5. L.N.G. Filon, Proc. Roy. Soc. (Edinburgh), A49, 38(1928).
6. Strang, G. y Fix, G.J., An Analysis of the Finite Element Method, Prentice Hall, Englewood Cliffs (1973).
7. Isaacson, E. y Keller, H.B., Analysis of Numerical Methods, Wiley, New York (1966).
8. W.R. Smith, D. Henderson y Y. Tago, J. Chem. Phys., 67 5308(1977).
9. L. Mier y Terán, E. Fernández Fassnacht y E. Quiñones, Physics Letters, 107A, 329(1985).

ESTA OBRA SE TERMINO DE IMPRIMIR EN EL MES DE OCTUBRE DE 1985, EN LA SECCION DE OFFSET DEL DEPARTAMENTO DE MATEMATICAS DEL CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL I.P.N.

LA EDICION CONSTA DE 500 EJEMPLARES Y SOBRANTES PARA REPOSICIÓN, SU CUIDADO ESTUVO A CARGO DEL SR. FILIBERTO MARTIGNON PEÑA.