14

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA EN CIENCIAS DE LA TIERRA

PROGRAMACIÓN AVANZADA APLICADA A LA INGENIERÍA PETROLERA

TESIS PROFESIONAL

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO PETROLERO

PRESENTA:

ROBERTO ARIEL GUZMÁN GUZMÁN

ASESOR: NÉSTOR MARTÍNEZ ROMERO

MÉXICO, D.F., CIUDAD UNIVERSITARIA 2001

UNIVERSIDAD NACIONAL AUTÓNOMA DE MEXICO

FACULTAD DE INGENIERIA
DIRECCION
60-I-818

SR. ROBERTO ARIEL GUZMAN GUZMAN
Presente

En atención a su solicitud, me es grato hacer de su conocimiento el tema que propuso el profesor M. en I. Néstor Martínez Romero y que aprobó esta Dirección para que lo desarrolle usted como tesis de su examen profesional de Ingeniero Petrolero:

PROGRAMACION AVANZADA APLICADA A LA INGENIERIA PETROLERA

Ruego a usted cumplir con la disposición de la Dirección General de la Administración Escolar en el sentido de que se imprima en lugar visible de cada ejemplar de la tesis el título de ésta.

Asimismo, le recuerdo que la Ley de Profesiones estipula que se deberá prestar servicio social durante un tiempo mínimo de seis meses como requisito para sustentar examen profesional

Atentamente
"POR MI RAZA HABLARA EL ESPIRITU"
Cd. Universitaria, D. F., a 9 de julio de 2001
EL DIRECTOR

ING. GERARDO FERRANDO BRAVO

GFB*RLLR*gtg

UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE INGENIERÍA

TESIS: **" PROGRAMACIÓN AVANZADA APLICADA A LA INGENIERÍA PETROLERA"**

PRESENTADA POR: ROBERTO ARIEL GUZMÁN GUZMÁN

DIRIGIDA POR: M. I. NÉSTOR MARTÍNEZ ROMERO

JURADO PARA EL EXAMEN PROFESIONAL

PRESIDENTE: ING. CARLOS LIRA SIL

VOCAL: M. I. NÉSTOR MARTÍNEZ ROMERO

SECRETARIO: M. I. JOSÉ ANTONIO GONZÁLEZ GUEVARA

1 ER SUPLENTE ING. EVA SÁNCHEZ OLEA

2 DO SUPLENTE M. I. JOSÉ MARTÍNEZ PÉREZ

MÉXICO, D. F., CD. UNIVERSITARIA, NOVIEMBRE 2001

AGRADECIMIENTOS

A Dios, por haberme dado lo mejor de la vida; la existencia, misma que me permitió el logro de una de mis grandes metas, el término de mi carrera profesional.

A mis padres, por todo su amor y paciencia que me han tenido para hacer posible un logro más; él cual no será el último, pero quizás el más importante. Gracias por la fe que depositaron en mi y por dármelo todo sin esperar a cambio más que el orgullo de hacer de mi un triunfador.

A mi hermana, como un testimonio de gratitud y correspondiendo al esfuerzo y al apoyo, que me permitieron continuar con mi superación profesional.

A toda mi familia, les agradezco por que de todos aprendí.

A mis amigos, Germán López Bautista, Hugo Hernández Ordóñez, Manuel A. Silva Romero, Ulises Neri Flores, por todo su apoyo y confianza que me han brindado y en especial a Iván Santamaría Vite y Oscar Osorio Peralta por todo el apoyo y la ayuda recibida para el desarrollo de este trabajo. A todos ustedes gracias por considerarme su amigo.

A mis amigas, Gabriela Araiza de la Rosa y Pilar Amieva por su amistad, cariño y apoyo que me han brindado.

Al Ing. Carlos Lira Sil, por sus valiosas observaciones para la realización final de este trabajo.

Al Ing. Néstor Martínez Romero, por darme su confianza para poder realizar este trabajo y por brindarme parte de su valioso tiempo.

Al Ing. José Antonio González Guevara, por el apoyo e interés recibido para la realización de este trabajo, así como la atención brindada hacia mi persona.

A la Ing. Eva Sánchez Olea, por su valiosa amistad y muy sinceramente por el apoyo, interés, y tiempo dedicado en la realización y revisión de este trabajo.

Al Ing. José Martínez Pérez, por la atención prestada a la revisión de este trabajo.

A la Universidad Nacional Autónoma de México, mi alma mater, por abrirme sus puertas, misma que me permitió el logro de una de mis grandes metas, estudiar una carrera profesional, así como el de brindarme la oportunidad de ser parte de tan importante y prestigiada institución.

A la Facultad de Ingeniería, por otorgarme el conocimiento a través de mi estancia en ella y el de haberme formado como Ingeniero.

Gracias Totales

Roberto Ariel Guzmán Guzmán

ÍNDICE GENERAL

ÍNDICE | FIGURAS

Figura Página

CAPÍTULO VI

CAPÍTULO VII

ÍNDICE | TABLAS

Tabla Página

CAPÍTULO IV

CAPÍTULO V

CAPÍTULO VI

RESUMEN

Este trabajo representa un material de apoyo para la asignatura de Programación Avanzada que se imparte dentro del plan de estudios de la carrera de Ingeniero Petrolero; el apoyo va dirigido tanto al profesor como a los estudiantes ya que presenta una cantidad importante de algoritmos programados en computadoras personales.

Se presentan conceptos y definiciones básicas, métodos numéricos, técnicas de programación para el cálculo y solución de sistemas de ecuaciones lineales y no lineales que se generan en los diferentes problemas del ámbito petrolero.

Se incluyen varios programas desarrollados en Microsoft Visual Basic® 6.0 los cuales dan solución a diversos problemas comunes en la Industria Petrolera como son: la solución del flujo en medios porosos para la simulación de los yacimientos, el diseño de un sistema de recolección y distribución de gas, el cálculo de los coeficientes del modelo de optimización de la perforación y el cálculo de la pseudo-presión de los gases reales, entre otros.

También se incluyen varias presentaciones hechas en Microsoft PowerPoint® como apoyo visual para la impartición de la asignatura de Programación Avanzada.

INTRODUCCIÓN I

I.1 BREVE HISTORIA DE LAS COMPUTADORAS.

Debido a la necesidad de almacenamiento de información y a su procesamiento, se vió la necesidad de idear formas que permitieran llevar un control más preciso de la información, así como un procesamiento más rápido de la misma.

Unos de los primeros instrumentos utilizados fue el ÁBACO, el cual se originó en medio oriente y es una calculadora decimal completa y manual.

La primera máquina de calcular fue construida por Whihelm Schickard (1542-1625), la cual podía sumar, restar, multiplicar y dividir, pero se perdió en la guerra de los años treinta.

Blas Pascal (1623-1662) recibe usualmente el crédito como creador de la primera máquina calculadora, la cual sólo podía sumar y restar.

Gottfried Wilhelm Leibniz (1646-1716) mejoró el invento de Pascal y soñó con el día en que todo razonamiento se pudiera efectuar dándole vuelta a la manivela.

A mediados del siglo XVIII se inventó un sistema para representar en tarjetas perforadas los dibujos de los tejidos. Los viejos telares manuales leían directamente las tarjetas, pero en 1810, Joseph Marie Jacquard inventó en Francia un telar mecánico provisto de una lectora automática de tarjetas. Las tarjetas se introducían en la máquina y salía en la tela el dibujo de colores.

Por otra parte, al otro lado del canal de la mancha, en Gran Bretaña, la idea de Jacquard originó una "reacción en cadena" en el cerebro de Charles Babbage (1792-1871), a quien se le conoce como padre de la computadora.

Babbage, profesor de matemáticas en Cambrige, había trabajado durante varios años en una enorme calculadora mecánica, a la que le había dado el nombre de "máquina de diferencias". Las tarjetas perforadas de Jacquard originaron una nueva idea genial a Babbage, creó una máquina a la que le llamó "máquina analítica". Las instrucciones de este mecanismo se leerían a través de las tarjetas perforadas.

Para ello la máquina necesitaría un dispositivo para la entrada de información, utilizándose una lectora de tarjetas. Finalmente, se necesitaba transferir los resultados del proceso a un dispositivo externo, es decir, a una unidad de salida. Babbage, diseñó la primera máquina tipográfica automatizada capaz de imprimir los resultados de los cálculos.

Por otra parte, surgieron las máquinas perforadoras de tarjetas, comenzando con las tabuladoras de datos censales diseñadas por Hollerith (1860-1929) inspirado en el telar de Jacquard, como lo había estado Babbage, Hollerith inventó una máquina con la única finalidad de acumular y clasificar información.

Hollerith fundó una compañía para fabricar sus máquinas procesadoras de datos operadas con tarjetas y encontró varios clientes; una compañía ferroviaria usó el sistema para conocer las estadísticas de los fletes. Otro cliente que encontró Hollerith fue un fabricante de herramientas quien dedicó las máquinas procesadoras a la compilación de costos, al análisis de nómina y a la administración del inventario.

Así la compañía de Hollerith obtuvo resultados financieros muy satisfactorios; tiempo después ingresó al campo de las calculadoras automáticas y de nueva cuenta logró un notable éxito, a esta compañía en la actualidad se le conoce como IBM (International Business Machines).

La tabuladora de Hollerith empleaba electricidad. En todo el complicado laberinto de circuitos, algunos investigadores centraron su atención en el dispositivo más sencillo de todos, "El interruptor". Un interruptor o switch es un elemento que puede abrir o cerrar un circuito eléctrico.

Las compañías telefónicas comenzaron a emplear el relevador, el cual, al recibir una señal eléctrica, se cierra y "releva" o envía su llamada al sitio correcto. Sin embargo el citado relevador no se acerca ni siquiera a otro tipo de "interruptor" inventado con anterioridad, nos referimos al "tubo electrónico de vacío". Un tubo electrónico de vacío también abre y cierra como un interruptor, a tal velocidad que uno ni siquiera puede ver parpadear la luz que emite. Poco tiempo después, se pensó en emplear estos dispositivos para sumar, almacenar e inclusive comprender relaciones lógicas.

El primer hombre que construyó la computadora electromecánica fue Konrad Zuse (1910-ND). Su dispositivo funcionaba por medio de relevadores, y leía la información de entrada en película perforada.

Pero en realidad las computadoras nacieron con la Segunda Guerra Mundial, ya que en los EE.UU. la marina colaboró con la Universidad de Harvard y la empresa IBM en la producción de la Mark I, gigante "electromagnetomecánico" que "nació" en 1944 diseñada por el profesor Howard Aiken; a espaldas de la marina, el ejército de EE.UU. destinó fondos a un proyecto de construcción de una computadora, sólo que en él se emplearían "tubos electrónicos de vacío". Los ingenieros a cargo del proyecto del ejército fueron J. Presper Eckert y John Manchly, el resultado fue una máquina calculadora del tamaño de un enorme granero, que se denominó "ENIAC: Electronic Numerical Integrater And Calculator" (Calculadora e Integradora Numérica Electrónica); "ENIAC" contenía unos 18,000 tubos electrónicos de vacío y realizaba 500 multiplicaciones por segundo.

Fue en el año de 1947, un año después de que se terminó "ENIAC", cuando un equipo de trabajo de la Universidad de Stanford inventó el "Transistor", en el que emplearon elementos conocidos con el nombre de "Semiconductores". A semejanza de los tubos electrónicos de

vacío, los transistores pueden hacer las veces de "Interruptores" pero son más pequeños, de operación más rápida, generan menos calor y tienen una vida más larga, además de que consumen menos energía eléctrica.

Posteriormente el transistor empezó a mostrar una increíble capacidad de disminución de tamaño y precio. Primero llegaron los "circuitos integrados", los cuales consisten en una tabla entera de transistores inter-construidos como una sola unidad, y después las integraciones en gran escala y en muy grande escala; (Large-Scale Integration, LSI, Very Large-Scale Integration, VLSI); en que se agrupan a cientos de miles de transistores en una pequeña pastilla llamada "CHIP".

Es en la década de los 60's cuando aparece la primera mini-computadora, la cual era aproximadamente del tamaño de un escritorio. En la década de los 70's surgieron la micro-computadoras que pueden ser tan pequeñas como uno quiera. En la década de los 80's a las computadoras grandes, se les denomina maxi-computadoras (Mainframes) y alcanzan una enorme capacidad.

I.2 ÚLTIMOS AVANCES TECNOLÓGICOS DE LA COMPUTACIÓN.

A principios de la década pasada y hasta ahora, la tecnología se ha estado desarrollando de manera gigantesca debido a grandes descubrimientos que dan origen a la aparición de nuevos y mejores materiales para construir los componentes electrónicos, por todo esto y más, en este periodo de cambios, la evolución de la computación ha sido enorme, así como el perfeccionamiento de los *microprocesadores* que cada vez son más eficientes, pequeños y rápidos, de las *memorias* y *discos duros* que tienen un mejor desempeño y velocidad de acceso, esto ocasionó que la computadora pasara de ser una herramienta que sólo las empresas e instituciones las obtuvieran y que sólo unas cuantas personas podían adquirirlas para uso personal, a ser una herramienta fundamental para cualquier tipo de disciplina llegando hasta una gran parte de hogares para el uso particular e incluso considerando que las *computadoras portátiles (laptop)* también han evolucionado enormemente.

Es por toda esta tecnología que ha salido al mercado en estos últimos años que estamos viviendo tiempos en que la interacción hombre-computadora es tan común en la vida cotidiana de las personas, y esto parte debido al exitoso surgimiento del *Internet*, el cual ha crecido en forma exponencial en sólo un par de años y que ha conseguido cambiar parte de nuestra forma de vida, ya que ahora con tan sólo tener acceso al *Internet* podemos estar en contacto con cualquier persona(s) de cualquier país por medio de una cuenta de *correo electrónico*, por el famoso *"Chat"*, por una video conferencia, o simplemente obtener algún tipo de información que requiramos y sólo por el costo de una llamada local.

Si, si es cierto que por un lado se encarece tener esta tecnología, por otro se abarata por el gran número de establecimientos que la ofrecen ya que por ganarse a los clientes bajan los costos.

Pero toda esta tecnología no funcionaría si no es por los numerosos avances que han tenido los *"lenguajes de programación"* (*software*). Estos lenguajes son una serie de instrucciones precisas que la computadora es capaz de entender y se basan en la utilización de reglas establecidas y acordadas para así poder tener una comunicación con la computadora y que se lleva a cabo mediante los recursos que ofrecen los lenguajes de programación.

Con la evolución de la computación han surgidos muchos *lenguajes de programación* con el fin de que el usuario dispusiera de una gran variedad de herramientas de programación para que le permitiesen sacar el máximo beneficio a la computadora. Los cuales iniciaron con los *binarios*, después fueron los *simbólicos o ensambladores*, luego llegaron los de *alto nivel* hasta los más recientes lenguajes de programación orientados a objetos, que son una serie de herramientas que permiten construir aplicaciones combinando fragmentos de programas ya prefabricados (*objetos gráficos*).

I.3 SITUACIÓN ACTUAL DE LA COMPUTACIÓN APLICADA A LA INGENIERÍA PETROLERA.

La ingeniería petrolera como una área de la ciencia aplicada, no se puede quedar al margen de toda esta tecnología que es la *"Computación"*, ya que los fenómenos físicos a los que se enfrenta debido a su complejidad involucran un sinnúmero de variables, que si no fuera por las potentes computadoras no podríamos estudiarlos ni darles solución. Es por eso que, para manejar esta enorme cantidad de información requerimos utilizar la mejor y más reciente tecnología, como son los poderosos microprocesadores que nos permiten hacer los complejos cálculos que se requieren para la solución de estos problemas, así como una gran capacidad tanto en memoria como en el disco duro, y por ende los tan especializados y numerosos *"softwares"*, como lo son los simuladores que son herramientas indispensables para visualizar el comportamiento de un fenómeno variante o varios de parámetros, es decir, se construyen modelos informáticos que describen el comportamiento de un sistema de interés, permitiendo la simulación de diversas condiciones de trabajo para apoyar la toma de decisiones, sin necesidad de construir modelos físicos que requieren tiempo, dinero y esfuerzo. Con la aplicación de estas herramientas se ahorra tiempo y dinero, ya que los resultados son casi inmediatos y no como con los modelos físicos, que se tenía que esperar semanas o meses para saber los resultados y así poder tomar una decisión.

I.4 ¿PORQUE UTILIZAR MICROSOFT VISUAL BASIC 6.0®?

La palabra *"BASIC"* hace referencia al lenguaje BASIC (Beginners All-Purpose Symbolic Instruction Code; o sea Código de Instrucciones Simbólicas Generales para Principiantes), es un lenguaje utilizado por más programadores que ningún otro lenguaje en la historia de la informática o computación. Visual Basic ha evolucionado a partir del lenguaje BASIC original y ahora contiene centenares de instrucciones, funciones y palabras clave, muchas de las cuales están directamente relacionadas con la interfase gráfica de Windows. Los principiantes pueden crear aplicaciones útiles con sólo aprender unas pocas palabras clave, pero, al mismo tiempo, la eficacia del lenguaje permite a los profesionales acometer cualquier objetivo que pueda alcanzarse mediante cualquier otro lenguaje de programación de Windows.

Para entender lo que es este lenguaje de programación contestaremos a la siguiente pregunta: ¿Qué es Visual Basic? La palabra *"Visual"* hace referencia al método que se utiliza para crear la interfase gráfica de usuario (GUI). En lugar de escribir numerosas líneas de código para describir la apariencia y la ubicación de los elementos de la interfaz, simplemente se agregan los objetos prefabricados en su lugar dentro de la pantalla. Si se ha utilizado alguna vez un programa de dibujo como Paint, ya se tiene la mayor parte de las habilidades necesarias para crear una interfaz de usuario efectiva.

El entorno de trabajo en Visual Basic 6.0® se denomina frecuentemente como Entorno Integrado de Desarrollo o IDE, ya que integra muchas funciones diferentes como el diseño, modificación, compilación y depuración en un entorno común. En las herramientas de desarrollo más tradicionales, cada una de esas funciones funcionan como un programa diferente, cada una con su propia interfaz.

Si el objetivo es crear un pequeño programa para uso personal o para un grupo de trabajo, un sistema para una empresa o incluso aplicaciones distribuidas de alcance mundial a través de Internet, Visual Basic 6.0® dispone de las herramientas que se necesitan.

Microsoft Visual Basic 6.0®, es la manera más rápida y sencilla de crear aplicaciones para el entorno Microsoft Windows®, tanto si es un profesional experimentado como un principiante en la programación, Visual Basic proporciona un juego completo de herramientas que facilitan el desarrollo rápido y sencillo de aplicaciones. Cabe mencionar que como todo lenguaje de programación no se exenta de errores internos.

La presente tesis tiene como objetivo principal el proporcionar un material de apoyo para la asignatura de Programación Avanzada, en la que se exponen técnicas de programación eficientes para el máximo aprovechamiento de las computadoras personales para su aplicación en las diferentes áreas de la Ingeniería Petrolera como lo son: Producción, Perforación y Yacimientos. También se incluye un CD de consulta el cual contiene todos los programas a los que hace referencia este trabajo, además contiene varias presentaciones

en Microsoft PowerPoint® para ser utilizadas como ayuda visual en la exposición de la asignatura (revisar Apéndice A).

A continuación se describe brevemente el contenido de los capítulos que conforman esta tesis.

Capítulo II, *Conceptos Básicos de Programación*. Detalla los conocimientos fundamentales sobre los conceptos de computación para entender el funcionamiento de los diferentes equipos de cómputo que se emplean, así como los diferentes tipos de errores que se generan al programar.

Capítulo III, *Solución de Sistemas de Ecuaciones* Lineales y No Lineales. Describe los diferentes métodos numéricos que facilitan de una manera rápida y eficiente la solución de los sistemas de ecuaciones lineales y no lineales que se generan al resolver un problema de Flujo a través de medios porosos, el de un Diseño de un sistema de recolección y distribución de gas, así como, al Diseñar una torre de perforación.

Capítulo IV, *Interpolación y Aproximación Numéricas*. Presenta algunas técnicas de programación para lograr soluciones prácticas y rápidas de fenómenos o procesos físicos y químicos, cuando la posibilidad de interpolar los datos o de generar una función de aproximación se torna primordial.

Capítulo V, *Integración Numérica*. Muestra los diferentes métodos matemáticos para la integración de funciones definidas en forma tabular obtenidas de algún experimento para así dar solución a las integrales involucradas en el cálculo de algunos problemas como puede ser el de la Pseudo-presión de los gases reales.

Capítulo VI, *Aproximación a la Solución de Ecuaciones Diferenciales*. Describe las diferentes técnicas numéricas más usuales para la solución de ecuaciones diferenciales ordinarias y parciales que representan fenómenos físicos como los que se originan principalmente en el área de Caracterización de Yacimientos.

Capítulo VII, *Técnicas Avanzadas de Programación con Aplicación a la Ingeniería Petrolera.* Presenta algunos conceptos de uso frecuente en la programación de computadores dentro de la Industria Petrolera. También se presentan los programas de cómputo SIMPP (Simulador de Presión) y REGRESML (Modelo de Regresión Múltiple), que son utilizados en la simulación del comportamiento de pruebas de presión en pruebas de decremento y en el análisis de regresión lineal, respectivamente.

Capítulo VIII, Conclusiones y Recomendaciones. De acuerdo con todo lo descrito en los capítulos anteriores, se realizan una serie de conclusiones y recomendaciones.

CONCEPTOS BÁSICOS DE PROGRAMACIÓN

II

II.1 INTRODUCCIÓN.

Un conocimiento sólido de conceptos de computación es primordial para entender el funcionamiento de los diferentes computadores que se emplean actualmente y para aquellos que aparecerán en un futuro próximo.

A pesar de los múltiples esfuerzos tecnológicos, las computadoras no son máquinas "perfectas", como la mayoría de la gente cree, debido a que sólo permiten el manejo de cierto rango de números reales, los cuales son representados con una cantidad determinada de cifras, incurriéndose, inevitablemente, en errores diversos.

La finalidad de este capítulo es describir dichos errores y generar en la actitud del lector un afán por evitarlos, para que así, al emplear equipos de cómputo en la solución de problemas reales, obtenga resultados con rapidez y precisión.

II.2 CONCEPTOS FUNDAMENTALES.

Una *computadora digital* es una máquina que procesa la información representada mediante combinaciones de señales discretas arregladas en forma codificada para representar datos en forma de caracteres o números. Específicamente, es un dispositivo para efectuar operaciones aritméticas y lógicas bajo el control de un programa almacenado.

Cuando una computadora ejecuta un programa, ésta interpreta una larga serie de unos (encendido) y ceros (apagado) escritos en *lenguaje máquina*. Si se tuviera que escribir un programa en lenguaje máquina, se tardaría bastante tiempo y resultaría fácil cometer errores.

Los lenguajes de programación como *Visual Basic*, *Fortran*, etc. son denominados lenguajes de alto nivel. Estos lenguajes sirven como intérpretes entre el lenguaje humano y el de la máquina intentan llenar el vacío entre los lenguajes humanos y el lenguaje de máquina. Los lenguajes avanzados, usualmente, tienen menos de doscientas palabras. Comparando esto con los lenguajes humanos como el Inglés, la mayoría de los cuales contienen alrededor de unas 100,000 palabras diferentes.

Estos lenguajes de programación no sólo están limitados a un número de palabras específicas, sino también al orden en que éstas pueden utilizarse. La *sintaxis* de un lenguaje define las palabras y el orden en que pueden utilizarse. La *semántica* define los significados de las palabras y sus relaciones. Cada lenguaje de programación dispone de reglas de semántica estrictas que limitan más los tipos de palabras que puede utilizar en las diferentes sentencias.

La *variable* es probablemente el concepto más importante a comprender antes de comenzar a programar. Una variable es un nombre que se le da a una posición de memoria. No importa qué posición de memoria utiliza la computadora, mientras que cada vez se utilice un mismo nombre de variable en el programa, la computadora utilizará la misma posición de memoria. De esta forma, una variable representa más a menudo un "valor que podría ser cualquier cosa" en lugar de un valor "desconocido".

La computadora realiza todos sus cálculos utilizando sólo ceros y unos. Los lenguajes de programación le permiten manipular esos ceros y unos directamente a través de la aritmética binaria.

Un dígito binario 1 (encendido) ó 0 (apagado) constituye una unidad básica denominada bit. La información se representa en las computadoras por grupos de bits, los que caracterizan

no solamente a números binarios sino también a otros símbolos discretos cualesquiera, sean éstos, dígitos decimales o letras del alfabeto.

A diferencia de los números decimales, también conocidos como sistema de *base diez* (dígitos que van del cero al nueve), los números binarios usan un sistema de *base dos* (sólo pueden ser cero o uno). Por ejemplo, el número binario $(111010)_2$ o también (111010b), donde la "b" al final significa que éste es binario, representa una cantidad que puede transmutarse a un número decimal, multiplicando cada bit por la base 2 elevada a una potencia entera, de la manera siguiente:

$$(1x2^5) + (1x2^4) + (1x2^3) + (0x2^2) + (1x2^1) + (0x2^0) = 58$$

Entonces, los seis bits $(111010)_2$, caracterizan un número binario cuyo equivalente decimal es 58, $(58)_{10}$; sin embargo, estos bits podrían representar un código binario para una letra del alfabeto o un código de control para especificar alguna decisión lógica.

La información binaria se representa por cantidades físicas, nombradas *señales*. Las señales eléctricas, como el voltaje, existen a través del sistema digital en dos formas reconocibles, que representan a una variable binaria igual a 1 (encendido) ó 0 (apagado). Las terminales de entrada de un circuito digital aceptan señales binarias dentro de tolerancias permisibles y los circuitos, en las terminales de salida, emiten respuestas con señales binarias que también cumplen con dichas tolerancias.

Una unidad de memoria es una colección de registros de almacenamiento, junto con los circuitos necesarios para transferir información dentro y fuera de los mismos. Si se puede tener acceso a los registros de memoria para realizar transferencia de datos cuando sea requerido (lectura/escritura), se habla de una Memoria de Acceso Aleatorio, RAM (*Random Access Memory*); si sólo es factible la obtención de información (lectura), entonces se hace mención de una Memoria de Solamente Lectura, ROM (*Read Only Memory*), en la cual, la información normalmente es creada por el fabricante.

Una unidad de memoria almacena información en grupos de bits, denominados *palabras* (*words*). Así, una *palabra* es una entidad de "n" bits. A una palabra de ocho bits comúnmente se le asigna el nombre de *byte*. Es costumbre referirse al número de bytes en una unidad de memoria con la literal *Kb*, la cual representa 1024 bytes de esta manera $1Kb = 1024$ bytes y $64Kb = 2^{16}$. El rango más común de memoria RAM es de 1024 a 2^{20} bytes. La Tabla 2.1 esboza una comparación entre estos conceptos informáticos y conceptos cotidianos.

CONCEPTO EN INFORMÁTICA	EQUIVALENCIA PRÁCTICA
8 bits = 1 byte	Nada equiparable
1 byte = 1 carácter	Cualquier letra, número, símbolo o espacio entre estos.
3 Kb = 3072 bytes	1 página (formato carta)
1 Megabyte = 1×10^6 bytes	Aprox. 333 páginas
1 Gigabyte = 1×10^9 bytes	Aprox. 350,000 páginas

Tabla 2.1. COMPARACIÓN ENTRE CONCEPTOS INFORMÁTICOS Y CONCEPTOS COTIDIANOS.

II.3 NÚMEROS EN PUNTO FLOTANTE.

Se han propuesto diferentes métodos para representar al conjunto infinito de números reales mediante el empleo de sistemas finitos de cómputo.

El método que actualmente se emplea en la gran mayoría de las computadoras es el denominado "*Números en Punto Flotante*", que emplea la técnica exponencial mediante el cual un número "*x*" puede obtenerse con la siguiente expresión:

$$x = \pm\left(\frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \ldots\ldots + \frac{d_t}{\beta^t}\right)\beta^e \quad , \tag{2.1}$$

donde:

$$d_1, d_2, \ldots\ldots, d_t, \text{son enteros que cumplen con: } d_1 > 0, \quad 0 \le d_i \le \beta - 1, \quad i = 1, \ldots\ldots, t$$

y: $L \le e \le U$; siendo '*e*', un número entero

La expresión (2.1) conforma un conjunto *F* de números, caracterizado por cuatro parámetros: un número base "β", un número de precisión o dígitos de la mantisa "*t*" y un rango del exponente (Límites de exponente) [*L, U*].

Cuando *x* es diferente de cero, $x \in F$, y $d_1 \neq 0$, se dice que *x* está normalizado. El entero ***e***, es llamado exponente, β^e es la parte entera y la fracción *f*, está dada por la expresión:

$$f = \left(\frac{d_1}{\beta} + \frac{d_2}{\beta^2} + ... + \frac{d_t}{\beta^t} \right) \quad . \tag{2.2}$$

La Tabla 2.2 muestra los valores usuales de los parámetros de la representación en punto flotante, empleados por diversas compañías. La columna β^{1-t} representa un valor estimado de la precisión aritmética del sistema (*Machine Epsilon*), la cual será tratada más adelante.

COMPUTADORA	S/D [a]	β	t	L	U	β^{1-t}
Honeywell 6000	S	2	27	-128	127	1.49×10^{-08}
Control Data 6600	D	2	48	-975	1070	7.11×10^{-15}
Burroghs b5500	D	8	13	-51	77	1.46×10^{-11}
Hewlett Packard HP-45	S	10	10	-98	100	1.00×10^{-09}
Texas Instruments SR-5x	D	10	12	-98	100	1.00×10^{-11}
IBM 360	S	16	6	-64	63	9.54×10^{-07}
IBM 370	D	16	14	-64	63	2.22×10^{-16}
Intel 8087	S	2	24	-126	63	6.96×10^{-08}
Intel 8087	D	2	53	-1022	1023	1.11×10^{-16}

Tabla 2.2. VALORES DE LOS PARÁMETROS DE LA REPRESENTACIÓN EN PUNTO FLOTANTE[22].

El conjunto de números en punto flotante, *F*, es finito y discontinuo; los elementos que contiene están dados por:

$$2(\beta - 1)\,\beta^{t-1}\,(U - L + 1) + 1 \quad . \tag{2.3}$$

[a] **Nota: S/D; Simple o Doble Precisión.**

Además no están igualmente espaciados a lo largo de su rango de valores. Aplicando la fórmula anterior, es posible estimar la cantidad de elementos que puede representar exactamente una computadora, por ejemplo:

$$F_{IBM\ DOBLE\ PRECISIÓN} \approx 1.729382257 \times 10^{19}\ números$$

$$F_{HP\ 45} \approx 3.582 \times 10^{12}\ números$$

II.4 ERRORES DE PRECISIÓN O REDONDEO.

Si suponemos un sistema en punto flotante (computadora hipotética), con los siguientes parámetros: $\beta = 2$, $t = 3$, $L = -1$ y $U = 2$, F estará integrado por 33 elementos, por tanto, no es posible que todos los números reales puedan ser representados y almacenados con este sistema, entonces cada número en F representa un intervalo de números reales. Si x representa un número real, contenido en un cierto intervalo del conjunto F, entonces al número en F más próximo a x se le denotará como $fl(x)$ (Figura 2.1).

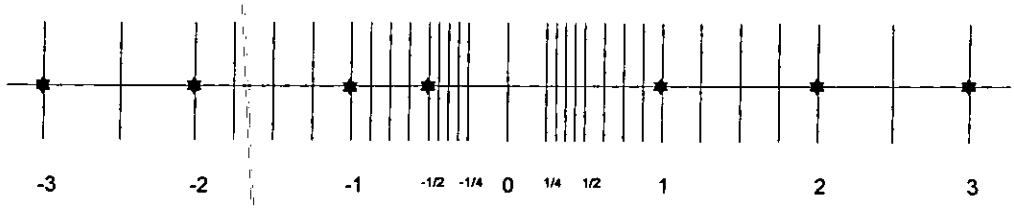


Figura 2.1. REPRESENTACIÓN GRÁFICA DE LOS NÚMEROS REALES QUE PUEDEN SER REPRESENTADOS.

El error relativo en que se incurre, es función de β y t:

$$\left|\frac{fl(x)-x}{x}\right| = \frac{1}{2}\beta^{1-t} \quad . \tag{2.4}$$

En sistemas de punto flotante con $\beta = 2$, o igual a una potencia de 2, al sumar 10 elementos de tamaño 0.1 no se obtiene como resultado al número 1.0, puesto que 0.1 no tiene representación finita en potencias de ½. Matemáticamente, esto es:

$$\frac{1}{10}=\frac{0}{2^1}+\frac{0}{2^2}+\frac{0}{2^3}+\frac{1}{2^4}+\frac{1}{2^5}+\frac{0}{2^6}+\frac{0}{2^7}+\cdots$$

Así, empleando subíndices para denotar la base β:

$$(0.1)_{10}=(0.000110011001100...)_2$$
$$=(0.012121212...)_4$$
$$=(0.063146314...)_8$$
$$=(0.199999999...)_{16}$$

Los elementos del lado derecho de las expresiones anteriores han sido truncados después de "t" dígitos, y al sumar diez de ellos, el resultado arrojado no ha sido la unidad.

Al sumar dos números en punto flotante ($x \oplus y$), donde $\oplus$ representa la operación suma en punto flotante, comúnmente se tiene como resultado una cantidad que no está en *F*; el valor real de la suma se puede aproximar por $fl(x + y)$. Lo ideal es que si ($x \oplus y$) cae dentro del rango de *F*, entonces $(x + y) = (x \oplus y) = fl(x + y)$, lo cual, en la gran mayoría de las computadoras se cumple sólo para algunos valores de "*x*" y de "*y*".

El *error por precisión o redondeo*, está representado por la diferencia numérica, en valor absoluto, entre ($x + y$) y ($x \oplus y$). Análogamente, puede calcularse en la resta, la multiplicación o la división.

En el sistema en punto flotante *F* supuesto con anterioridad se observa que:

$$x=\frac{5}{4} \quad , \quad y=\frac{3}{8} \quad \in F$$

$$\frac{5}{4}\oplus\frac{3}{8} \quad \approx \quad \frac{3}{2} \text{ ó } \frac{7}{4} \quad \neq \quad \frac{5}{4}+\frac{3}{8}=\frac{13}{8} \quad \text{(suma real)}$$

La razón por la cual $\frac{5}{4}+\frac{3}{8}$ no está en el conjunto *F*, se debe al espaciamiento de sus elementos. Ahora, si se desea realizar la suma $\frac{7}{2}+\frac{7}{2}=7$, no estaría en *F*, ya que el número siete sería mayor que el número más grande del conjunto, presentándose entonces una señal de *overflow*, interrumpiéndose en la mayoría de las computadoras el proceso de cálculo. En la multiplicación ($x \bullet y$) se encuentra con mayor frecuencia esta señal, puesto que esta operación abarca $2t$ ó $2t-1$ dígitos significativos.

Por otro lado, una señal de *underflow* suele desplegarse cuando se multiplican dos números ($x \bullet y$), diferentes de cero, cuyo resultado es más pequeño que el menor número, también diferente de cero, representado en *F*. Este desplegado, aunque es inusual, se presenta aún en la suma. Las operaciones de adición y de multiplicación en punto flotante no son distributivas ni asociativas, pero sí conmutativas.

Los errores de precisión se generan al emplear en los cálculos, series o números irracionales $(\sqrt{2},\pi,e,\text{etc.})$, ya que existe la imposibilidad de representarlos exactamente en la memoria de la computadora, entonces, al efectuarse alguna operación se usa la aproximación más cercana.

II.4.1 MACHINE EPSILON.

La precisión de la suma en punto flotante puede determinarse por medio del *Machine Epsilon*, esto es, el número más pequeño (ε) tal que:

$$(1 \oplus \varepsilon) > 1 \quad . \tag{2.5}$$

Existen algunos métodos para calcular el valor (o una aproximación) de ε. A continuación se presenta un programa, en lenguaje *Visual Basic*, que permite el cálculo aproximado del *Machine Epsilon* de un equipo de cómputo, así como la ventana de resultado (ver figura 2.2).

```
Sub Calcular_Click ()
Dim MEP As Double
MEP = 1
10  T = 1 + MEP
    If T > 1 Then
      MEP = MEP / 2: GoTo 10
    End If
Label1.Caption = MEP
End Sub
```

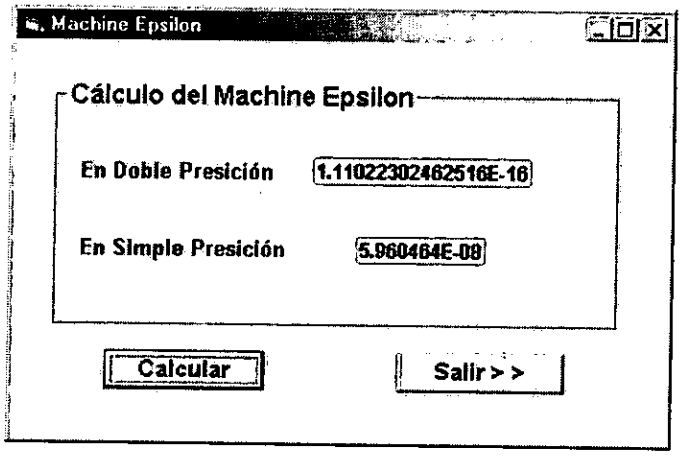


Figura 2.2. VENTANA DE RESULTADOS DEL MACHINE EPSILON (ε) PARA SIMPLE Y DOBLE PRESICION.

Este concepto define la precisión y la calidad del computador, esto es, entre menor sea el valor de ε, mayor será la precisión del procesador del equipo empleado. En la Tabla 2.3 se concentran los valores de ε, para algunos equipos de cómputo.

EQUIPO DE CÓMPUTO	MACHINE EPSILON (ε)
PC AT ZEOS 486SLC (80486)	5.960464×10^{-08}
Micro Computadora Casio FX-795P	$5.820766091 \times 10^{-11}$
PC Acer Plus (80486) 66 MHz	0.111022×10^{-15}
PC HP AMD Athlon® 1.1 GHz	$1.11022302462516 \times 10^{-16}$
PC AMD Athlon®1.33 GHz	$1.11022302462516 \times 10^{-16}$
PC HP Intel Pentium®III 900 MHz	$1.11022302462516 \times 10^{-16}$

Tabla 2.3. VALORES DEL MACHINE EPSILON (ε) DE ALGUNAS COMPUTADORAS.

II.4.2 EJEMPLO DE ERROR POR REDONDEO.

Supóngase que desea evaluar la función e^x mediante un programa de cómputo. Tal función puede representarse como la sumatoria de una serie convergente infinita, es decir:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots \quad . \tag{2.6}$$

Si el sistema de cómputo está caracterizado por $\beta = 10$ y $t = 5$, y se necesita el valor de e^{-x}, para $x = 5.5$, entonces, sustituyendo en la serie anterior se tendrán para cada término los resultados siguientes:

$e^{-5.5} =$ 1.0000
-5.5000
+15.125
-27.730
+38.129
-41.942
+38.446
-30.208
+20.768
-12.692
+6.9803
-3.4902
+1.5997
.
.
0.0026363

El resultado final contempla sólo 25 términos, puesto que, los términos subsecuentes no lo modifican. ¿Es satisfactoria esta respuesta? No, ya que el resultado correcto es:

$$e^{-5.5} = 0.00408677 \quad .$$

Nótese que algunos términos, y consecuentemente las sumas intermedias, son mucho más grandes en magnitud que la respuesta final; por ejemplo, el quinto término, 38.129, involucra

en sí mismo un error por redondeo tan considerable como el del resultado final, debido a que el cuarto dígito decimal se ha perdido. Por el impacto que tiene esa pérdida sobre el resultado, al fenómeno anterior se le denomina *cancelación catastrófica*.

Una solución podría ser el emplear un mayor número de cifras significativas, aunque esto generaría un incremento en el tiempo de ejecución y a menudo requiere de técnicas de programación un tanto sofisticadas. Sin embargo, una solución práctica es la de calcular e^{-x}, para $x = 5.5$ y luego obtener su valor recíproco, es decir:

$$e^{-5.5} = \frac{1}{e^{5.5}} = \frac{1}{1+5.5+15.125+\cdots} = 0.0040865 \quad .$$

Obsérvese, que el resultado anterior reduce el error relativo a un 0.007% y además contiene 5 cifras significativas. Se puede concluir que el cálculo de ciertas operaciones, empleando una computadora, no debe ser excesivamente complicado para incurrir en grandes errores por redondeo.

Mediante técnicas adecuadas de programación siempre es posible reducir al mínimo los errores de redondeo.

II.5 ERRORES INHERENTES, DE TRUNCAMIENTO Y DE APROXIMACIÓN FUNCIONAL.

Los modelos matemáticos de procesos físicos o naturales, siempre introducen *errores inherentes*, los cuales, son el resultado de una inadecuada o parcial interpretación del fenómeno natural a representar, de la aleatoriedad del mismo y de la falta de certeza de las mediciones experimentales. A menudo, un modelo incluye únicamente los rasgos más sobresalientes del proceso físico y es deliberadamente despojado de detalles superfluos, que son catalogados como de segundo término.

Aquellos algoritmos que usan sólo operaciones aritméticas y ciertas operaciones lógicas, como lo son las comparaciones algebraicas, son llamados *Métodos Numéricos*. El error que

se origina al aproximar la solución de un problema matemático a través de un método numérico que considera series infinitas es, usualmente denominado *error de truncamiento*.

El aproximar una función *f(x)*, por medio de una función "conveniente", *g(x)*, promueve la aparición del error de *aproximación funcional*. Error que frecuentemente se presenta en dos casos. El primero, es reemplazar una función *f(x)*, difícil de evaluar o manipular (por ejemplo: diferenciales o integrales), por una expresión, *g(x)*, más simple. El segundo caso, es la interpolación de valores tabulados de funciones. La función *f(x)* es, cuantitativamente conocida para un número finito de argumentos, llamados *puntos base*, es decir:

$$x_0 , f(x_0)$$
$$x_1 , f(x_1)$$
$$\cdots\cdots$$
$$\cdots\cdots$$
$$x_i , f(x_i)$$
$$\cdots\cdots$$
$$\cdots\cdots$$
$$x_n , f(x_n)$$

Entonces, debe generarse la función de aproximación, *g(x)*, que estime el valor de *f(x)*, para $x \neq x_i$, $i = 0,1,\ldots,n$.

La efectividad de una función *g(x)*, y por ende una disminución del error de aproximación funcional, depende de varios factores, entre ellos están los siguientes: el conocimiento de la función original, el origen y la precisión de los valores tabulados, y la precisión ofrecida por la aproximación empleada.

II.6 PROBLEMAS RESUELTOS.

Problema Resuelto II.1. La serie de Taylor para calcular la función error es:

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{n!(2n+1)} \quad ,$$

la cual converge para todo valor de *x*. Escriba un programa que evalúe *erf(x)* usando esta serie, considerando tantos términos como sea necesario, de manera que el primero en descartarse no altere considerablemente la suma acumulada. Como esta serie es alternante, el error causado por truncar la suma infinita debe ser mayor que el error de redondeo. Estime el efecto del error de truncamiento, comparando el valor calculado de *erf(x)* con uno obtenido de tablas o de una subrutina publicada. Considere *x* = 0.5, 0.8,1, 2, 3, 5, 8 y 10. Explique los resultados.

Sugerencia:

Forsythe[1] propone un ingenioso ciclo iterativo, en lenguaje *Fortran* que se ha emigrado a *Visual Basic*, a utilizar en parte del desarrollo de su programa:

```
DO
   OLDS = S
   EN = EN + 1.0
   T = -XSQ * T * (2.0 * EN - 1.0) / (EN * (2.0 * EN + 1.0))
   S = S + T
LOOP WHILE OLDS <> S
```

¿Qué valores deben asignarse a *T*, *S*, *EN* y *XSQ* antes de iniciar con el ciclo? No olvide el factor $\frac{2}{\sqrt{\pi}}$.

Solución:

Empleando la serie de *Taylor* "pura", esto es, sin modificaciones, se obtuvieron los siguientes resultados ver Tabla 2.4 (revisar anexo, consultar Apéndice A):

x	erf(x)	NÚMERO DE TÉRMINOS
0.5	0.520500280905103	4
0.8	0.74209482748	5
1.0	0.842699296675635	7
2.0	0.99531079671	15
3.0	1.0000015616	28
5.0	0.999978525315654	71
8.0	10853600567.889	170*
10.0	$-7.19504274137654 \times 10^{+34}$	154*

Tabla 2.4. RESULTADOS DE erf(x) PARA LA SERIE DE TAYLOR.

Ahora, para utilizar el algoritmo propuesto por *Forsythe*, deben determinarse previamente, los valores iniciales de las variables involucradas. Si se desarrolla la serie de *Taylor*, se tiene:

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}}\left[x - \frac{x^3}{1!\,3} + \frac{x^5}{2!\,5} - \frac{x^7}{3!\,7} + \frac{x^9}{4!\,9} - \ldots\ldots\right] .$$

Por otro lado, para el algoritmo:

$$EN = 1 \Rightarrow T_1 = \frac{-XSQ^2(T)}{1!\,3} ,$$

$$EN = 2 \Rightarrow T_2 = \frac{-XSQ^2(T_1)}{2!\,5}(1!\,3) ,$$

$$EN = 3 \Rightarrow T_3 = \frac{-XSQ^2(T_2)}{3!\,7}(2!\,5) .$$

* MÁXIMO NÚMERO DE ITERACIONES PERMITIDAS POR EL EQUIPO DE CÓMPUTO UTILIZADO, SIN INCURRIR EN UN ERROR DE OVERFLOW.

Para que los correspondientes sumandos sean iguales, los valores iniciales que se les asignarán deben ser: $EN = 0$, $S = X$, $T = X$ y $XSQ = X^2$. Se recomienda al lector la demostración de lo anterior.

Los resultados del uso del artificio se presentan a continuación (revisar el de consulta):

x	erf(x)	NÚMERO DE ITERACIONES
0.5	0.520499877779065	12
0.8	0.742100964659211	15
1.0	0.842700792894698	18
2.0	0.995322264953972	31
3.0	0.999977909437718	48
5.0	1.00000002184711	95
8.0	1670464837.07938	183
10.0	$-1.1546774143543 \times 10^{+24}$	243*

Tabla 2.5. RESULTADOS DE erf(x) PARA EL ALGORITMO DE FORSYTHE.

Los resultados arrojados por ambos procedimientos son muy semejantes para valores del argumento iguales o inferiores a la unidad, sin embargo, su alternancia induce errores mayúsculos para valores grandes de *x*, provocados por cancelación. Los dos métodos presentan cierta dispersión con respecto a los valores tabulados para *erf(x)*. Consúltese la Tabla 2.6.

x	erf(x)
0.5	0.52049
0.8	0.74210
2.0	0.99532
3.0	0.99870
3.9	1.00000

Tabla 2.6. RESULTADOS DE VALORES TABULADOS PARA erf(x).

* MÁXIMO NÚMERO DE ITERACIONES PERMITIDAS POR EL EQUIPO DE CÓMPUTO UTILIZADO, SIN INCURRIR EN UN ERROR DE OVERFLOW.

Problema Resuelto II.2. ¿Qué respuesta se obtiene al ejecutar el programa en *Visual Basic* mostrado a continuación, en una computadora? Explique por qué.

```
'Problema Resuelto II.2
X = 0.0 : H = 0.1!
FOR I = 1 TO 10
X = X + H
NEXT I
Y = 1.0 - X
Label1.Caption = X : Label2.Caption = Y
```

Respuesta:

Al ejecutar el programa anterior en una computadora personal, con "X" y "Y" en *doble precisión* y "H" en *simple precisión*, se obtiene el siguiente resultado:

X = 1.000000014901161000000
Y = 0.000000014901161193848

El valor lógico a obtener para la variable "*X*" debía ser el de la unidad y cero para la variable "*Y*", pero debido a la representación en punto flotante que posee la computadora en uso, se tendrán valores que tiendan a la unidad y a cero respectivamente. En este caso el valor de *y* representa el error de precisión absoluto, cometido al calcular "*X*".

Ahora, si se toman "X", "Y" y "H" en *doble precisión*, se obtiene el siguiente resultado:

X = 1.0
Y = 0.00000000000000011102302462516

Siguiendo con el mismo esquema ahora se toman "X" y "Y" en *simple precisión* y "H" en *doble precisión*, obteniéndose el siguiente resultado:

X = 1.0

Y = -0.00000011920930

Problema Resuelto II.3. ¿Es posible representar exactamente el número decimal 0.1 en su computadora? Si no, ¿cuál es el número, en punto flotante, más cercano? ¿Cuál es el número menor, también en punto flotante, más cercano? ¿Qué valor es asignado en cada una de las siguientes expresiones? Datos: $x = 0.1$, $y = 0.1$. Use el siguiente programa en lenguaje *Visual Basic*:

```
'Problema Resuelto II.3
'Valores en simple precisión
X = 0.1 : Y = 0.1
Label1.Caption = X : Label2.Caption = Y
'Valor en doble precisión
Y = 0.1# : X = 0.1#
Label3.Caption = Y
X = 1.0/10.0 'Resultado de una operación en simple precisión
Y = 1.0#/10.0# 'Resultado de una operación en doble precisión

Label4.Caption = X : Label5.Caption = Y
'Valores en notación científica
X = 1.0E-1 : Y = 1.0D-1
Label6.Caption = X : Label7.Caption = Y
'Valores de lectura
X = Val(Text1.Text) : Y = Val(Text2.Text)
Label8.Caption = X : Label9.Caption = Y
```

Resultados:

Al ejecutar el programa anteriormente presentado, se obtienen los siguientes valores, ver Tabla 2.7:

	X	Y
Simple precisión	0.1	0.1000000014901161
Doble precisión	0.1	0.1
X en Simple, Y en Doble	0.1	0.1
Notación Científica	0.1	0.1
Valores leídos	0.1	0.1

Tabla 2.7. RESULTADOS DE LA REPRESENTACIÓN DEL NÚMERO DECIMAL 0.1.

La computadora empleada representa con una exactitud de ocho decimales al número 0.1. Los números en punto flotante más cercanos a 0.1, superior e inferiormente, son aproximadamente iguales a $0.1 \pm 1.4901161 \times 10^{-09}$.

Problema Resuelto II.4. ¿Cómo representa internamente su computadora a los números $^1/_2$, $^2/_3$ y $^3/_5$?

a) Use una notación apropiada, por ejemplo, binaria, octal o hexadecimal.

b) Considere el siguiente programa en lenguaje *Visual Basic*:

```
'Problema Resuleto II.4
H = 1.0/2.0
X = 2.0/3.0 - H
Y = 3.0/5.0 - H
E = ( X + X + X ) - H
F = ( Y + Y + Y + Y + Y ) - H
Q = F / E
Label1.Caption = "H =" & H
Label2.Caption = "X =" & X
Label3.Caption = "Y =" & Y
Label4.Caption = "E =" & E
Label5.Caption = "F =" & F
Label6.Caption = "Q =" & Q
```

La variable Q puede asumir diferentes valores que dependerán del punto flotante del hardware aritmético usado por la computadora. Trate de estimar el valor de Q para computadoras con las que esté familiarizado. Corra el programa en tantos equipos como le sea posible para corroborar los resultados. Explíquelos.

Resultados:

a) Empleando subíndices parar denotar la base, se observa:

$$\left(\frac{1}{2}\right)_{10} = (0.5)_{10} = (0.1)_2 = (0.1)_8 = (0.1)_{16} \quad ,$$

$$\left(\frac{2}{3}\right)_{10} = (0.666666)_{10} = (0.101010)_2 = (0.525252)_8 = (0.AAAAAA)_{16} \quad ,$$

$$\left(\frac{3}{5}\right)_{10} = (0.6)_{10} = (0.10011001)_2 = (0.46314631)_8 = (0.999999)_{16} \quad .$$

Por lo tanto, como los números $^2/_3$ y $^3/_5$ no tienen una representación finita en los sistemas en punto flotante más usuales (bases 2, 8 y 16), ya que son números periódicos, se generarán errores de precisión, al usar éstos y otros números similares, en la solución de algún problema.

La Tabla 2.8 que se muestra a continuación, indica el número base utilizado en el sistema en punto flotante de equipos de cómputo de uso común.

MARCA DEL EQUIPO	NÚMERO BASE
IBM 360	16
Burroughs 6500	8
Univac 1108	2
Honeywell 6000	2
Intel Pentium® III	2
AMD Athlon®	2

Tabla 2.8. NÚMERO BASE UTILIZADO EN EL SISTEMA EN PUNTO FLOTANTE DE EQUIPOS DE CÓMPUTO DE USO COMÚN.

b) Empleando una computadora personal, y simple precisión para las variables *E*, *F* y *Q*; se tiene:

H = 0.5	E = 1.490116×10^{-08}
X = 0.1666667	F = 7.450581×10^{-09}
Y = 0.1	Q = 0.5

Contrariamente a lo esperado $Q \neq (0/0 = \infty)$, ya que se presentan errores de precisión y truncamiento, aunados al tipo de representación en punto flotante del equipo usado. Ahora, si se emplea doble precisión para todas las variables, se obtiene:

H = 0.5	E = $-2.775557561562891 \times 10^{-17}$
X = 0.1666666666666667	F = $2.775557561562891 \times 10^{-17}$
Y = 0.1	Q = $-1 \Rightarrow Q \neq (0/0 = \infty)$

Para tener un panorama más amplio de que tan alejados se está de la verdadera solución a continuación se muestran los resultados reales de éstas variables.

H = 1/2	E = 0
X = 1/6	F = 0
Y = 1/10	Q = ∞

Problema Resuelto II.5. Evalúe la siguiente serie infinita:

$$\Phi(x) = \sum_{k=1}^{\infty} \frac{1}{k(k+x)} \quad ,$$

para x = 0 a 1.0 con incrementos de 0.1, con un error inferior a 0.5×10^{-8}

Nota: Esto requiere emplear tanto el análisis humano como el poder de una computadora, ya que ni lo uno ni lo otro causaría éxito usándose individualmente. No desperdicie el tiempo de máquina de su computadora tratando de calcular la sumatoria sin razonar antes la posible solución (¿Cuánto tiempo podría desperdiciarse?).

Sugerencia:

Use el siguiente artificio[1]:

$$\frac{1}{k(k+1)} = \frac{1}{k} - \frac{1}{k+1} \quad ,$$

para probar que $\Phi(1) = 1$. Entonces usar, $\Phi(x) - \Phi(1)$ como una serie que converge más rápido que la que define a $\Phi(x)$. Se tendrá que repetir este truco antes de obtener la serie para calcular $\Phi(x)$ y así lograr una convergencia rápida.

Solución:

En una computadora personal, sin utilizar el artificio sugerido, se obtuvieron los resultados que se muestran en la Tabla 2.9. (revisar el ⊛ de consulta):

x	Φ(x)	ITERACIONES	TIEMPO [seg]
0.0	1.64486336299137	14143	0.0
0.1	1.53453654129765	14143	0.0
0.2	1.44080813818977	14143	0.046875
0.3	1.36001187867615	14142	0.0
0.4	1.28950709293472	14142	0.0625
0.5	1.22734057015388	14142	0.0
0.6	1.17203448876863	14142	0
0.7	1.12244863542935	14142	0.0546875
0.8	1.07768816588781	14142	0.0
0.9	1.0370402112442	14142	0.0546875
1.0	0.999929293643498	14142	0.0

Tabla 2.9. RESULTADOS DE LA SERIE PROPUESTA SIN UTILIZAR EL ARTIFICIO DE FORSYTHE.

Usando el artificio de la sugerencia:

Ahora, sustituyendo el valor de $x = 1$ en la serie infinita propuesta, además, aplicando propiedades de las series, se tiene:

$$\Phi(1) = \sum_{k=1}^{\infty} \frac{1}{k(k+1)} = \sum_{k=1}^{\infty} \frac{1}{k} - \sum_{k=1}^{\infty} \frac{1}{k+1} \quad .$$

Las series del lado derecho de la ecuación, son series conocidas, de las cuales se sabe que:

$$\sum_{k=1}^{\infty} \frac{1}{k} = 2 \quad ,$$

$$\sum_{k=1}^{\infty} \frac{1}{k+1} = 1 \quad ,$$

entonces, reemplazando estos valores en la serie $\Phi(1)$:

$$\Phi(1) = \sum_{k=1}^{\infty} \frac{1}{k(k+1)} = 2 - 1 = 1 \quad .$$

Con lo que se prueba que $\Phi(1) = 1$.

Por otro lado, expresando $\alpha(x) = \Phi(x) - \Phi(1)$, se observa lo siguiente:

$$\begin{aligned} \alpha(x) &= \Phi(x) - \Phi(1) \\ &= \sum_{k=1}^{\infty} \frac{1}{k(k+x)} - \sum_{k=1}^{\infty} \frac{1}{k(k+1)} \\ &= \sum_{k=1}^{\infty} \left[\frac{1}{k(k+x)} - \frac{1}{k(k+1)} \right] \\ &= \sum_{k=1}^{\infty} \left[\frac{k - kx}{k^3 + k^4 + k^2 x + k^3 x} \right] \end{aligned} \quad .$$

Se deduce que la expresión resultante convergerá más aprisa que la serie original, puesto que en su denominador existen elementos elevados a potencias mayores que 1.

La manera más fácil de calcular $\Phi(x)$ será entonces, calcular la serie $\alpha(x)$ y sumarle $\Phi(1) = 1$. Los resultados arrojados, usando este artificio se presentan en la Tabla 2.10 (revisar el ⊛ de consulta anexo).

x	Φ(x)	ITERACIONES	TIEMPO [seg]
0.0	1.64493260997507	585	0.0
0.1	1.53460583955566	565	0.0
0.2	1.44087748940656	543	0.0
0.3	1.36008129207236	519	0.0
0.4	1.28957657129892	493	0.0
0.5	1.22741012155786	464	0.0
0.6	1.17210412461756	431	0.0
0.7	1.12251836671268	391	0.0
0.8	1.07775802325446	342	0.0
0.9	1.03711024269184	271	0.0
1.0	1.00000000000000	1	0.0

Tabla 2.10. RESULTADOS DE LA SERIE PROPUESTA UTILIZANDO EL ARTIFICIO DE FORSYTHE.

Se concluye que la rapidez con la cual una computadora puede ofrecer resultados, se incrementa cuando el análisis humano depura anticipadamente el proceso a ejecutar, es decir, no basta con poseer la computadora más sofisticada y costosa del mercado, si no se dispone de analistas de cómputo capaces de conformar una buena mancuerna.

II.7 PROBLEMAS PROPUESTOS.

Problema Propuesto II.1. Algunas funciones matemáticas pueden ser difíciles de calcular, además, al emplearse diversas aproximaciones para resolver un mismo problema, los resultados pueden ser disímbolos. Para ejemplificar estas aseveraciones, realice un programa de cómputo que evalúe la función error por tres métodos, que enseguida, se discutirán, para argumentos en el rango de $0 \le x \le 5$, a intervalos de 0.5. Dicha función está definida como:

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \quad .$$

En aquellos métodos que consideren series infinitas, encuentre el resultado usando 5 y 10 términos. Utilice simple precisión en todos sus cálculos. Compare los resultados con los ofrecidos por alguna subrutina disponible o por alguna tabla de *erf(x)*; incluya sus observaciones al respecto.

Método 1.

La serie de Taylor puede utilizarse para representar a la función exponencial e^x. Entonces, sustituyéndola en la función error, se tiene:

$$\operatorname{erf}(x)=\frac{2}{\sqrt{\pi}}\int_0^x\left(1-t^2+\frac{t^4}{2!}-\frac{t^6}{3!}+.....\right)dt \quad ,$$

por lo que, integrando, se obtiene:

$$\operatorname{erf}(x)=\frac{2}{\sqrt{\pi}}\left(x-\frac{x^3}{(3)1!}+\frac{x^5}{(5)2!}-\frac{x^7}{(7)3!}+.....\right) \quad .$$

Como puede observarse, la serie anterior es alternante, por lo cual es inexacta para valores grandes del argumento, ya que existirán errores substanciales por cancelación.

Método 2.

Si se integra por partes la función *erf(x)*, se logra la siguiente serie:

$$\operatorname{erf}(x)=\frac{2xe^{-x^2}}{\sqrt{\pi}}\left(1+\frac{2x^2}{1(3)}+\frac{(2x^2)^2}{1(3)(5)}+\frac{(2x^2)^3}{1(3)(5)(7)}+.....\right) \quad .$$

Esta serie no es alternante. ¿Qué opina de su eficiencia?

Método 3.

Una función racional de aproximación (ajuste funcional), en ocasiones brinda mayor exactitud cuando se usa el mismo número de coeficientes comparada con la serie original. Tal aproximación, para el caso presentado, podría ser:

$$\mathrm{erf}(x) = \frac{1}{\left(1 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4\right)^4} + \epsilon(x) \quad ,$$

donde:

$$a_1 = 0.278393$$
$$a_2 = 0.230389$$
$$a_3 = 0.000972$$
$$a_4 = 0.078108$$
$$|\varepsilon(x)| \leq 5 \times 10^{-4}$$

¿Es esta aproximación más exacta que las dos anteriores?.

SOLUCIÓN DE SISTEMAS DE ECUACIONES LINEALES Y NO LINEALES

III

III.1 INTRODUCCIÓN.

El concepto de simulación de un sistema, es decir, la simulación del comportamiento de un sistema, sea éste, por ejemplo, un yacimiento o una red de distribución de gas, tiene su base en el desarrollo y operación de un modelo físico y/o matemático, cuyo comportamiento refleje ante condiciones cualesquiera el del sistema considerado.

Un modelo matemático es una ecuación, o un conjunto de ellas, que simula los procesos físicos que ocurren en un sistema al variar una o todas las condiciones iniciales. La Simulación Numérica de Yacimientos Petroleros, que es ejemplo de la aplicación de un modelo matemático, hace uso de métodos numéricos para lograr la solución de la(s) ecuación(es) que constituye(n) al sistema.

El objetivo del presente capítulo, es mostrar técnicas numéricas simples que faciliten la solución de sistemas de ecuaciones lineales y no lineales.

III.2 SISTEMAS DE ECUACIONES LINEALES.

Un sistema de ecuaciones lineales es un conjunto de ecuaciones con la forma siguiente:

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & & \vdots & & \vdots & & \vdots & \vdots & \vdots \\ a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n & = & b_m \end{array}, \qquad (3.1)$$

que en forma matricial, puede representarse como:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}. \qquad (3.2)$$

Empleando notación algebraica, se tiene: $A\bar{x} = \bar{b}$, donde A representa a la matriz de coeficientes, $\bar{x}$ al vector columna de incógnitas y $\bar{b}$ al vector columna de términos independientes; a la matriz (A, $\bar{b}$), de orden *"m × (n+1)"*, se le asigna el nombre de *matriz ampliada del sistema*.

El grupo de *"n"* valores $x_1, x_2,\ldots,x_n$ que satisfacen simultáneamente a todas las ecuaciones, es la solución del sistema. Dicha solución puede tipificarse en tres casos, éstos son:

- Solución única, se dice entonces que el sistema es *compatible o consistente, y determinado*. La matriz A es *no-singular*.
- Solución múltiple, se tiene un sistema *compatible (consistente) e indeterminado*. La matriz A es *singular*.
- El sistema no admite solución, por lo que se le nombra *incompatible o inconsistente*. La matriz A es *singular*.

Es bien conocido que el determinante de una matriz coadyuva a la detección de su singularidad (si el det A ≠ 0, A es no singular), pero, al emplear equipos de cómputo en la solución de sistemas de ecuaciones, se generan errores (consultar Capítulo II), que pueden provocar la emisión equivocada de criterios sobre el problema a resolver. Para esclarecer tal

aseveración, supóngase una matriz diagonal de tamaño 100 x 100, cuyos elementos diagonales son todos iguales a 0.1. Es evidente que su determinante es distinto de cero (0.1^{100}), no obstante, este valor es tan pequeño que cualquier computador arrojaría un valor de cero a pesar de que la matriz en cuestión es no singular. Debido al problema anterior, es conveniente introducir el concepto de número de condición de una matriz (Cond), el cual refleja de manera más eficaz que su determinante, la cercanía a la estabilidad o no singularidad. Matemáticamente, el número de condición se denota como:

$$\mathrm{Cond}(A) = \frac{\max\limits_{x} \dfrac{\|A\bar{x}\|}{\|\bar{x}\|}}{\min\limits_{x} \dfrac{\|A\bar{x}\|}{\|\bar{x}\|}} \geq 1 \quad . \tag{3.3}$$

El símbolo "‖ ‖" representa la norma o módulo de un vector. Es usual que el número de condición se exprese también de la siguiente forma:

$$\mathrm{Cond}(A) = \frac{\lambda_{máx}}{\lambda_{mín}} \geq 1 \quad , \tag{3.4}$$

donde:

λ = Valor característico o Eigen-valor.

Si A es una matriz cuadrada (n × n), sus valores característicos pueden obtenerse empleando el Método de las Potencias[4], basado en la relación $A\bar{x} = \lambda\bar{x}$, o encontrando las raíces del polinomio:

$$\det (A - \lambda I) = 0 \, , \tag{3.5}$$

donde:

I = matriz identidad.

Si Cond = 1, la matriz es no singular (estable) y el sistema de ecuaciones tendrá solución única, entonces un número de condición mucho mayor que la unidad indicará inestabilidad o singularidad, en este caso, con una pequeña variación en alguno de los coeficientes de la matriz, se obtendrá un vector $\bar{x}$ diferente del originalmente calculado (soluciones múltiples). En pocas palabras, el número de condición esboza la sensibilidad del vector solución $\bar{x}$, a cambios en los elementos de A y $\bar{b}$.

En seguida se detallarán las técnicas más empleadas en la solución de sistemas de ecuaciones lineales.

III.2.1 REGLA DE CRAMER.

Este método resuelve sistemas de ecuaciones lineales de tamaño (n x n). Si el determinante de la matriz A es diferente de cero (det A ≠ 0), el valor de la k-ésima incógnita se aclarará efectuando el cociente de los determinantes de las matrices A_k y A, donde A_k, se obtiene al reemplazar la k-ésima columna de la matriz de coeficientes (A), por el vector de términos independientes ($\bar{b}$).

En otras palabras, sea $A\vec{x} = \bar{b}$, un sistema de *"n"* ecuaciones lineales con *"n"* incógnitas y sea A = [$a_{i,j}$] su matriz de coeficientes. Si el det A ≠ 0, entonces:

$$x_k = \frac{\det A_k}{\det A} \quad , \quad k = 1, 2, ..., n \, , \tag{3.6}$$

donde:

$$A_k = \left[c_{i,j}\right] \quad , \quad c_{i,j} = \begin{cases} a_{i,j} & , \text{ para } j \neq k \quad ; \; i = 1, 2, ..., n \\ b_j & , \text{ para } j = k \quad ; \; j = 1, 2, ..., n \end{cases} . \tag{3.7}$$

La Regla de Cramer se tipifica como una técnica "exacta", puesto que no es iterativa. Si se contempla un equipo de cómputo que ejecute 10,000 multiplicaciones/seg, siendo "n" el orden del determinante, se requiere el número de operaciones y tiempos presentados en la Tabla 3.1.

n	NÚMERO DE MULT. APROX. n!	TIEMPO DE MÁQUINA
5	100	0.01 seg
12	5×10^8	10 horas
20	2×10^{18}	10 años
30	3×10^{32}	10^{21} años

Tabla 3.1. OPERACIONES Y TIEMPOS NECESARIOS CON LA REGLA DE CRAMER.

Se concluye que este método no es adecuado para resolver sistemas mayores de (4×4), pues se realizarán gran cantidad de operaciones (ya que implica el cálculo de determinantes), convirtiendo en *"lenta"* la obtención del vector solución.

III.2.2 MÉTODO DE ELIMINACIÓN GAUSSIANA.

El método de Gauss es utilizado para resolver sistemas de ecuaciones lineales del tipo $(m \times n)$. Consiste en obtener sistemas equivalentes, a partir del original, aplicando *transformaciones elementales* sobre la matriz ampliada del sistema, hasta llevarlo a la *forma escalonada* (matriz triangular superior), esto es, cuando el número de ceros anteriores al primer elemento no nulo de cada renglón aumenta al pasar de un renglón al siguiente, hasta llegar, eventualmente, a renglones cuyos elementos son todos nulos.

Las principales transformaciones elementales, son:

a) Intercambio de un renglón por otro.
b) Adición, término a término, de dos renglones.
c) Multiplicación de la totalidad de los elementos de un renglón por un escalar diferente de cero.

Una vez reducido el sistema original a la forma escalonada, la solución se logra, de manera directa, por sustitución hacia atrás ("x_i" se obtiene por simple despeje de la i-ésima ecuación, entonces se sustituye en la ecuación "i-1" y se calcula el valor de x_{i-1}, este procedimiento se continúa hasta tenerse los *"n"* valores de *"x"*).

Como este método no es de aproximaciones sucesivas, la solución debería ser exacta, sin embargo, debido a errores de redondeo o precisión, no lo es. Para minimizar el error, es recomendable seleccionar en cada ocasión como pivote al mayor elemento, en valor absoluto, del sistema[4] (se recomienda consultar el problema resuelto III.1, página 91).

Considerando una computadora digital que ejecute 10,000 multiplicaciones/seg, siendo *"n"* el orden del sistema, se tiene[6]:

n	NÚMERO DE MULT. APROX. $n^3/3$	TIEMPO DE MÁQUINA
5	40	0.005 seg
12	500	0.050 seg
20	2500	0.250 seg
30	9000	1.000 seg

Tabla 3.2. OPERACIONES Y TIEMPOS NECESARIOS CON EL MÉTODO DE ELIMINACIÓN GAUSSIANA.

Se observa que esta técnica no realiza una gran cantidad de operaciones, y tampoco invierte demasiado tiempo de máquina, aunque aumente el número de ecuaciones del sistema.

III.2.3 ALGORITMO DE THOMAS.

Se emplea para dar solución a sistemas de ecuaciones lineales de orden (n × n), cuya matriz de coeficientes presente un carácter "n-diagonal". Para efectos didácticos, considérese un sistema tri-diagonal, tal que:

$$A=\begin{bmatrix} b_1 & c_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & \vdots & \vdots & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & \vdots & \vdots & 0 & 0 \\ 0 & 0 & 0 & 0 & \vdots & \vdots & \vdots & 0 \\ 0 & 0 & 0 & 0 & 0 & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & 0 & 0 & 0 & 0 & 0 & a_n & b_n \end{bmatrix}.$$

Este algoritmo se basa en la *Técnica de Descomposición L-U de una matriz*, esto es:

$$A = L\,U\,. \tag{3.8}$$

Donde, L es una matriz triangular inferior y U es una triangular superior, o sea:

$$L=\begin{bmatrix} \alpha_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_2 & \alpha_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_3 & \alpha_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & \vdots & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \vdots & \vdots & 0 & 0 \\ 0 & 0 & 0 & 0 & \vdots & \vdots & 0 \\ 0 & 0 & 0 & 0 & 0 & a_n & \alpha_n \end{bmatrix} \quad , \quad U=\begin{bmatrix} 1 & \beta_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \beta_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \beta_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \vdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \vdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & \beta_{n-1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

El producto (L U) es:

$$LU=\begin{bmatrix} \alpha_1 & (\alpha_1\beta_1) & 0 & 0 & 0 & 0 & 0 \\ a_2 & (a_2\beta_1+\alpha_2) & (\alpha_2\beta_2) & 0 & 0 & 0 & 0 \\ 0 & a_3 & (a_3\beta_2+\alpha_3) & (\alpha_3\beta_3) & 0 & 0 & 0 \\ 0 & 0 & \vdots & \vdots & \vdots & 0 & 0 \\ 0 & 0 & 0 & \vdots & \vdots & \vdots & 0 \\ 0 & 0 & 0 & 0 & (a_{n-1}) & (a_{n-1}\beta_{n-2}+\alpha_{n-1}) & (\alpha_{n-1}\beta_{n-1}) \\ 0 & 0 & 0 & 0 & 0 & a_n & (a_n\beta_{n-1}+\alpha_n) \end{bmatrix}.$$

Como A = L U, cada elemento de (L U) será igual al de A, por lo cual:

$$\begin{aligned} a_i &= a_i && , \quad i = 2,3,.....,n \\ \alpha_1 &= b_1 \\ \alpha_i &= b_i - (a_i\beta_{i-1}) && , \quad i = 2,3,.....,n \\ \beta_i &= \frac{c_i}{\alpha_i} && , \quad i = 2,3,.....,(n-1) \end{aligned} \quad .$$

Entonces, un sistema de ecuaciones $A\bar{x} = \bar{f}$ puede representarse como (recordemos que A = L U):

$$LU\bar{x} = \bar{f} \tag{3.9}$$

Además, considerando que $U\bar{x} = \bar{y}$, se tendrá el sistema:

$$L\bar{y} = \bar{f} \ , \tag{3.10}$$

el cual puede resolverse fácilmente por sustitución hacia adelante. De manera que:

$$y_1 = \frac{f_1}{\alpha_1} \ , \tag{3.11}$$

$$y_i = \frac{f_i - (a_i y_{i-1})}{\alpha_i} \quad ; \quad i = 2, 3, ..., n \ . \tag{3.12}$$

Finalmente, el sistema U x = y, se soluciona mediante una sustitución hacia atrás, por lo tanto:

$$x_n = y_n \ , \tag{3.13}$$

$$x_i = y_i - (\beta_i \, x_{i+1}) \quad ; \quad i = (n-1), (n-2), ..., 1 \ . \tag{3.14}$$

Para una matriz penta-diagonal de orden (n × n) como la siguiente, se tiene:

$$A = \begin{bmatrix} c_1 & d_1 & 0 & e_1 & 0 & 0 & 0 & 0 \\ b_2 & c_2 & d_2 & 0 & e_2 & 0 & 0 & 0 \\ 0 & b_3 & c_3 & d_3 & 0 & e_3 & 0 & 0 \\ a_4 & 0 & b_4 & c_4 & d_4 & 0 & e_4 & 0 \\ 0 & \vdots & 0 & \vdots & \vdots & \vdots & 0 & \vdots \\ 0 & 0 & \vdots & 0 & \vdots & \vdots & \vdots & 0 \\ 0 & 0 & 0 & a_{n-1} & 0 & b_{n-1} & c_{n-1} & d_{n-1} \\ 0 & 0 & 0 & 0 & a_n & 0 & b_n & c_n \end{bmatrix} .$$

Mientras que las matrices L y U serán:

$$L=\begin{bmatrix} \alpha_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_2 & \alpha_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ \gamma_3 & \beta_3 & \alpha_3 & 0 & 0 & 0 & 0 & 0 \\ \delta_4 & \gamma_4 & \beta_4 & \alpha_4 & 0 & 0 & 0 & 0 \\ 0 & \vdots & \vdots & \vdots & \vdots & 0 & 0 & 0 \\ 0 & 0 & \vdots & \vdots & \vdots & \vdots & 0 & 0 \\ 0 & 0 & 0 & \vdots & \vdots & \vdots & \vdots & 0 \\ 0 & 0 & 0 & 0 & \delta_n & \gamma_n & \beta_n & \alpha_n \end{bmatrix} \qquad U=\begin{bmatrix} 1 & u_1 & v_1 & w_1 & 0 & 0 & 0 & 0 \\ 0 & 1 & u_2 & v_2 & w_2 & 0 & 0 & 0 \\ 0 & 0 & 1 & \vdots & \vdots & \vdots & 0 & 0 \\ 0 & 0 & 0 & 1 & \vdots & \vdots & \vdots & 0 \\ 0 & 0 & 0 & 0 & 1 & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 1 & u_{n-2} & v_{n-2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & u_{n-1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Con un desarrollo similar al anterior, se obtiene:

$$\delta_i = a_i \qquad , \quad i = 4,5,\ldots..,n$$

$$\gamma_3 = 0$$

$$\gamma_i = -\delta_i\, u_{i-3} \qquad , \quad i = 4,5,\ldots..,n$$

$$\beta_i = b_i \qquad , \quad i = 2,3$$

$$\beta_i = b_i - (\gamma_i\, u_{i-2}) - (\delta_i\, v_{i-3}) \qquad , \quad i = 4,5,\ldots..,n$$

$$\alpha_1 = c_1$$

$$\alpha_i = c_i - (\beta_i\, u_{i-1}) \qquad , \quad i = 2,3$$

$$\alpha_i = c_i - (\beta_i\, u_{i-1}) - (\gamma_i\, v_{i-2}) - (\delta_i\, w_{i-3}) \qquad , \quad i = 4,5,\ldots..,n$$

$$u_1 = \frac{d_1}{\alpha_1}$$

$$u_i = \frac{d_i - (\beta_i\, v_{i-1})}{\alpha_i} \qquad , \quad i = 2,3$$

$$u_i = \frac{d_i - (\beta_i\, v_{i-1}) - (\gamma_i\, w_{i-2})}{\alpha_i} \qquad , \quad i = 4,5,\ldots..,(n-1)$$

$$v_1 = 0$$

$$v_i = \frac{-\beta_i\, w_{i-1}}{\alpha_i} \qquad , \quad i = 2,3,\ldots..,(n-2)$$

$$w_i = \frac{e_i}{\alpha_i} \qquad , \quad i = 1,2,\ldots..,(n-3)$$

De igual manera, para los vectores "$\bar{y}$" y "$\bar{x}$":

$$y_1 = \frac{f_1}{\alpha_1}$$

$$y_2 = \frac{f_2 - (\beta_2\, y_1)}{\alpha_2}$$

$$y_3 = \frac{f_3 - (\beta_3\, y_2) - (\gamma_3\, y_1)}{\alpha_3}$$

$$y_i = \frac{f_i - (\beta_i\, y_{i-1}) - (\gamma_i\, y_{i-2}) - (\delta_i\, y_{i-3})}{\alpha_i} \quad , \quad i = 4,\ldots\ldots, n$$

$$x_n = y_n$$

$$x_i = y_i - (u_i\, x_{i+1}) \quad , \quad i = n-1$$

$$x_i = y_i - (u_i\, x_{i+1}) - (v_i\, x_{i+2}) \quad , \quad i = n-2$$

$$x_i = y_i - (u_i\, x_{i+1}) - (v_i\, x_{i+2}) - (w_i\, x_{i+3}) \quad , \quad i = (n-3), (n-4),\ldots\ldots, 1$$

Esta técnica es equivalente al método de Eliminación Gaussiana, y con ella se evita el incremento del error asociado a la solución regresiva de las ecuaciones y se atenúan los requerimientos de memoria RAM. Además, se realizan cuando mucho 5n multiplicaciones en comparación con $n^3/3$ en el de Gauss y n! en la Regla de Cramer, para resolver un sistema (n × n). En el de consulta se presentan programas de cómputo para la solución de matrices tri y penta-diagonales, empleando el Algoritmo de Thomas y la técnica de Descomposición L-U, respectivamente, Thomas.exe y LU-Penta.exe.

III.2.4 SUBRUTINAS DECOMP Y SOLVE[1].

Existen sistemas de cómputo que incluyen en sus programas de biblioteca, programas para obtener la solución de sistemas de ecuaciones lineales, los cuales, generalmente, están basados en el método de Eliminación Gaussiana. Tal es el caso de las subrutinas DECOMP y SOLVE, que serán tratadas a continuación.

A DECOMP le corresponde la primera etapa de la eliminación de la matriz ampliada del sistema (dejar ceros abajo de la diagonal principal), mientras que SOLVE usa esos cálculos, realizando una sustitución hacia atrás, para generar la solución del sistema.

Estas subrutinas incluyen la técnica de Pivoteo Parcial (se sugiere consultar el problema resuelto III.1, página 91), esto es, al k-ésimo paso de la primera etapa de la eliminación, se elige como pivote, al mayor elemento (en valor absoluto), contenido en la k-ésima columna no reducida. El renglón en que se ubica este pivote es intercambiado por el k-ésimo renglón, quedando el pivote en la posición (k, k). Similares intercambios se llevan a cabo en los elementos del vector de términos independientes.

Además, DECOMP estima un valor del número de condición (COND) de la matriz de coeficientes; si se detecta singularidad COND será igual a *1E+32*, en caso contrario, será *1*. Cualquier valor de COND entre *1* y *1E+32*, indicará la posición relativa de la matriz con respecto a la no-singularidad.

Si se desea evaluar el determinante de la matriz de coeficientes, bastará con efectuar el producto de algunos elementos calculados por DECOMP, o sea:

$$\det(A) = IPVT(N) \times a(1,1) \times a(2,2) \times \ldots \times a(N,N)\ , \qquad (3.15)$$

donde IPVT es el vector pivote obtenido por Decomp y a( ) son los elementos de la diagonal principal.

La inversa de la matriz de coeficientes (A), puede definirse como la matriz x, cuyas columnas $\bar{x}_j$, satisfacen:

$$A\bar{x}_j = \bar{e}_j \quad , \quad j = 1, 2, \ldots\ldots, n \quad , \qquad (3.16)$$

donde $\bar{e}_j$ es la j-ésima columna de la matriz identidad (I). Por lo tanto, para obtener la matriz inversa x, es necesario llamar a DECOMP una sola vez y a SOLVE *"n"* ocasiones, una para cada columna de x. Si se detecta singularidad, x será incalculable[1].

DECOMP y SOLVE son las subrutinas más eficientes para solucionar sistemas de ecuaciones lineales, pues no sólo ofrecen precisión en los resultados, sino además, un

tiempo de ejecución aceptable. En el CD de consulta, se presenta en lenguaje *Visual Basic*, estas subrutinas con el nombre de Decomp.bas, (consultar Apéndice A).

III.2.5 SOLUCIÓN AL PROBLEMA DE FLUJO BIDIRECCIONAL, MONOFÁSICO, TRANSITORIO, A TRAVÉS DE MEDIOS POROSOS.

La simulación de un yacimiento se hace, en la mayoría de los casos, con el fin de pronosticar su comportamiento al ser sometido a diferentes programas de explotación. El objetivo de esta sección, es elaborar un simulador numérico. Por lo cual, se desarrollan algunas técnicas para la solución de las ecuaciones involucradas en el modelo matemático que describe el flujo bidimensional, monofásico y transitorio a través de medios porosos.

La expresión que describe el flujo bidireccional, monofásico, en régimen transitorio (variable), de un fluido ligeramente compresible (agua o aceite) y de viscosidad constante, que fluye en un medio poroso homogéneo de espesor constante y anisótropo, es:

$$k_x \frac{\partial^2 p}{\partial x^2} + k_y \frac{\partial^2 p}{\partial y^2} + \frac{119.56\,\mu\, q}{h\,\Delta x\,\Delta y} = 119.56\,\phi\,\mu\, c \frac{\partial p}{\partial t} \quad , \tag{3.17}$$

donde:

x = desplazamiento en la coordenada x, [m].
y = desplazamiento en la coordenada y, [m].
k_x = permeabilidad en la dirección x, [mD].
k_y = permeabilidad en la dirección y, [mD].
t = tiempo, [días].
h = espesor del medio poroso, [m].
p = presión, [kg/cm^2 abs].
μ = viscosidad, [cp].
c = compresibilidad, [cm^2/kg].
q = gasto de fluido, a condiciones estándar, en función de espacio y tiempo (sí es de producción se usa con signo negativo), [m^3 /día].

Además, las condiciones de frontera (CF) e iniciales (CI) son:

$$CF\begin{cases} \text{interna} & p(0,x,t) = p_0 \\ & p(y,0,t) = p_0 \end{cases} \qquad (3.18)$$

$$CF\begin{cases} \text{externa} & p(L,x,t) = p_1 \\ & p(y,M,t) = p_1 \end{cases} \qquad (3.19)$$

$$CI\ \{p(x,y,0) = P(x,y) = \text{valor conocido},$$

donde:

L = longitud total en la dirección x, [m].

M = longitud total en la dirección y, [m].

Para resolver la ecuación (3.17), se aplica el método de Diferencias Finitas (el cual es tratado en el Capítulo VI, sección VI.3.1), teniéndose los esquemas Explícito, Implícito y de Crank-Nicholson, como alternativas de solución. Antes de abordar estas técnicas, se expondrán los conceptos de estabilidad y convergencia.

Los cálculo necesarios para resolver un problema determinado, dependerán del tiempo de predicción, las características del yacimiento y de los fluidos, los gastos de inyección o de producción, el tamaño de las celdas de la malla y la precisión que se desee en los resultados.

Al disminuir los incrementos Δx y Δy usados en la ecuación (3.29) Δt también disminuye y se reduce el error de truncamiento en la aproximación de las derivadas, pero aumenta el tiempo de cálculo. Por otra parte, al emplear Δt grandes, además de incrementarse el error de truncamiento, se tiene la posibilidad de que se presente oscilación en los resultados, fallando el método.

Por lo tanto, la Δt adecuada en cada problema se obtendrá haciendo un balance de los factores anotados con anterioridad.

III.2.5.1 CONVERGENCIA.

Si en un punto (x_i , t_n) de un determinado esquema, se cumple que:

$$\lim_{\Delta x, \Delta t \to 0} \left| p_i^n - p(x_i, t_n) \right| \approx 0 \quad , \tag{3.20}$$

$$i = 1, 2, \ldots\ldots, L \quad , \quad n = 1, 2, \ldots\ldots, t \quad ,$$

el esquema es convergente[3], es decir, tiende a la solución real. Gráficamente, esto se representa en la Figura 3.1.

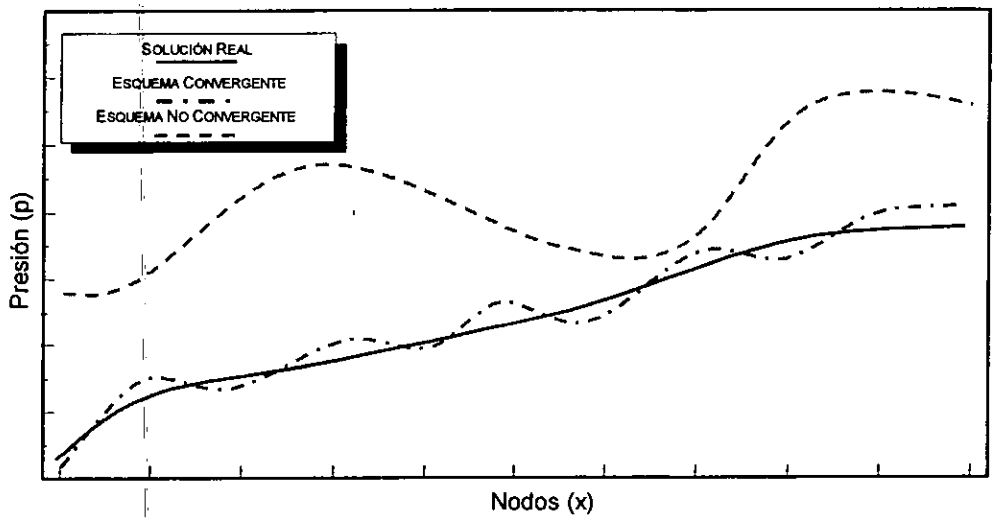


Figura 3.1. CONVERGENCIA, para t = n.

III.2.5.2 ESTABILIDAD.

Si en un esquema determinado, al mantener fijos los valores de los incrementos Δx y Δt, el error al aproximar la solución real no se amplifica mientras $\Delta t \to \infty$, se dice que es un esquema estable. Matemáticamente, se tiene:

$$\lim_{t \to \infty} \left| p_i^n - p(x_i, t_n) \right| < 1 \quad , \tag{3.21}$$

$$i = 1, 2,.....,L \quad , \quad n = 1, 2,.....,t \quad ,$$

En otras palabras si el valor calculado no varía considerablemente con respecto al real, se habla de un esquema estable. Esto se muestra en la Figura 3.2. Es importante apuntar que se tendrá estabilidad sólo si se presenta la convergencia.

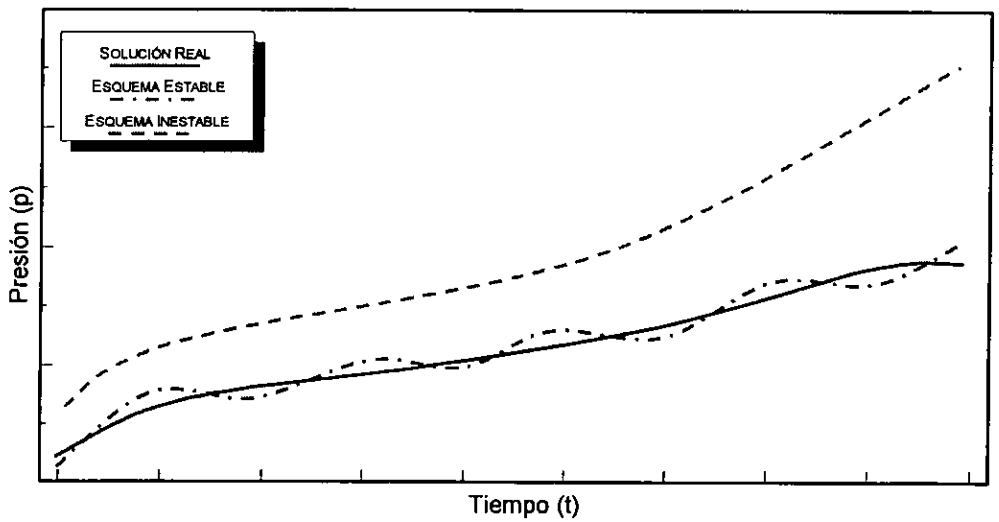


Figura 3.2. ESTABILIDAD, nodo i.

III.2.5.3 ESQUEMA EXPLÍCITO.

Partiendo de la ecuación (3.17), anteriormente expuesta, y considerando:

$$\beta_1 = \frac{119.56\,\mu\ q}{h\,\Delta x\,\Delta y} \quad , \tag{3.22}$$

$$\beta_2 = 119.56\,\phi\,\mu\,C_t \quad . \tag{3.23}$$

Se tiene:

$$k_x \frac{\partial^2 p}{\partial x^2} + k_y \frac{\partial^2 p}{\partial y^2} + \beta_1 = \beta_2 \frac{\partial p}{\partial t} \quad . \tag{3.24}$$

Ahora, aproximando las derivadas parciales de la ecuación (3.24) por medio de Diferencias Finitas del tipo centrales en posición y progresivas en tiempo, se tiene que:

$$\text{Diferencias finitas progresivas en tiempo}: \quad \left\{\frac{\partial p}{\partial t}\right\}^{n}_{i,j} = \frac{p^{n+1}_{i,j} - p^{n}_{i,j}}{\Delta t} \quad , \tag{3.25}$$

$$\text{Diferencias finitas centrales en posición}: \begin{cases} \left\{\dfrac{\partial^2 p}{\partial x^2}\right\}^{n}_{i,j} = \dfrac{p^{n}_{i-1,j} - 2p^{n}_{i,j} + p^{n}_{i+1,j}}{(\Delta x)^2} \quad , & (3.26) \\ \left\{\dfrac{\partial^2 p}{\partial y^2}\right\}^{n}_{j,j} = \dfrac{p^{n}_{i,j-1} - 2p^{n}_{i,j} + p^{n}_{i,j+1}}{(\Delta y)^2} \quad . & (3.27) \end{cases}$$

En las relaciones anteriores, el subíndice *"i"* representa a la distancia o nodos en la dirección *x* (abscisa), *"j"* a los nodos en la dirección *y* (ordenada), y el superíndice *"n"*, al tiempo.

Sustituyendo las ecuaciones (3.25), (3.26) y (3.27) en la (3.24), despejando $p^{n+1}_{i,j}$ y factorizando, se llega a la expresión siguiente:

$$p^{n+1}_{i,j} = \frac{\Delta t\, k_x}{\beta_2 (\Delta x)^2}\left(p^{n}_{i-1,j} + p^{n}_{i+1,j}\right) + \left[1 - 2\frac{\Delta t}{\beta_2}\left(\frac{k_x}{(\Delta x)^2} + \frac{k_y}{(\Delta y)^2}\right)\right] p^{n}_{i,j} + \frac{\Delta t\, k_y}{\beta_2 (\Delta y^2)}\left(p^{n}_{i,j-1} + p^{n}_{i,j+1}\right) + \frac{\Delta t\, \beta_1}{\beta_2} \quad , \tag{3.28}$$

$$i = 1, 2, \ldots\ldots, L \; ; \quad j = 1, 2, \ldots\ldots, M \; ; \quad n = 0, 1, \ldots\ldots, t$$

La ecuación (3.28) representa un esquema explícito, de sencilla solución, evaluado al tiempo *"n"* (t_n). Este esquema es inusual en la práctica, puesto que es condicionalmente convergente y por ende, condicionalmente estable, se puede demostrar matemáticamente que la convergencia se cumplirá sólo para[2]:

$$\Delta t \leq \frac{1}{2\left[(\Delta x)^{-2} + (\Delta y)^{-2}\right]} \; . \tag{3.29}$$

Por lo tanto, si la simulación de flujo a través de un medio poroso involucra grandes Δt, el riesgo de divergencia se hace patente.

Debido a que el empleo de Diferencias Finitas involucra una expansión en Serie de Taylor, se generan errores de truncamiento, ya que no se consideran todos los términos de la serie.

En la Figura 3.3 se presenta la configuración de la malla que define la técnica explícita, en la cual, se indican los nodos implicados en la solución de las diferenciales.

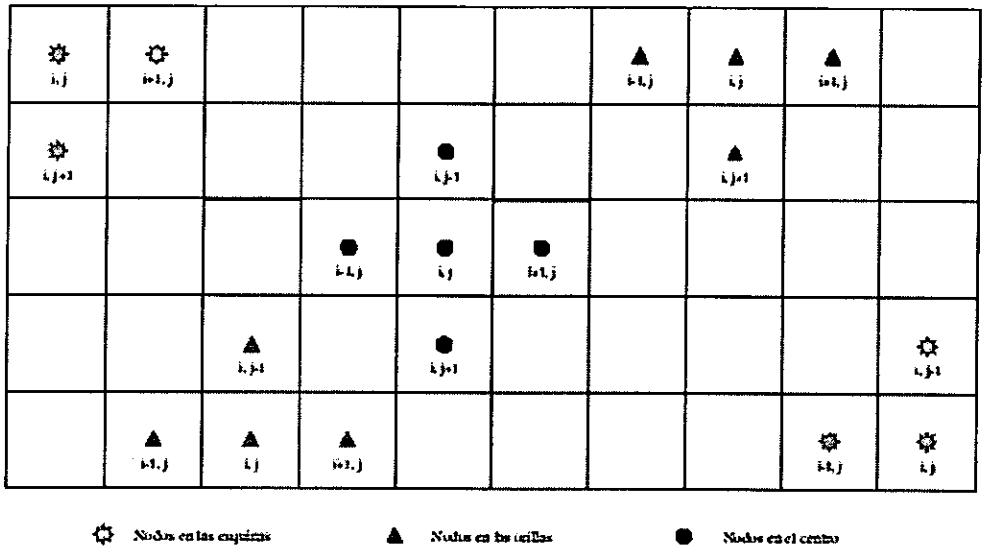


Figura 3.3. MALLA USADA EN EL ESQUEMA EXPLÍCITO, AL TIEMPO N.

III.2.5.4 ESQUEMA IMPLÍCITO.

Este esquema, a diferencia del explícito, se evalúa al tiempo *"n+1"* (t_{n+1}). Lo cual, puede ser apreciado en la malla de la Figura 3.4.

Conservando la nomenclatura y empleando Diferencias Finitas, se tiene entonces, para las derivadas parciales de la ecuación (3.24):

Diferencias finitas regresivas en tiempo: $$\left\{\frac{\partial p}{\partial t}\right\}_{i,j}^{n+1} = \frac{p_{i,j}^{n+1} - p_{i,j}^{n}}{\Delta t} \quad , \tag{3.30}$$

Diferencias finitas centrales en posición:

$$\left\{\frac{\partial^2 p}{\partial x^2}\right\}_{i,j}^{n+1} = \frac{p_{i-1,j}^{n+1} - 2p_{i,j}^{n+1} + p_{i+1,j}^{n+1}}{(\Delta x)^2} \quad , \tag{3.31}$$

$$\left\{\frac{\partial^2 p}{\partial y^2}\right\}_{i,j}^{n+1} = \frac{p_{i,j-1}^{n+1} - 2p_{i,j}^{n+1} + p_{i,j+1}^{n+1}}{(\Delta y)^2} \quad . \tag{3.32}$$

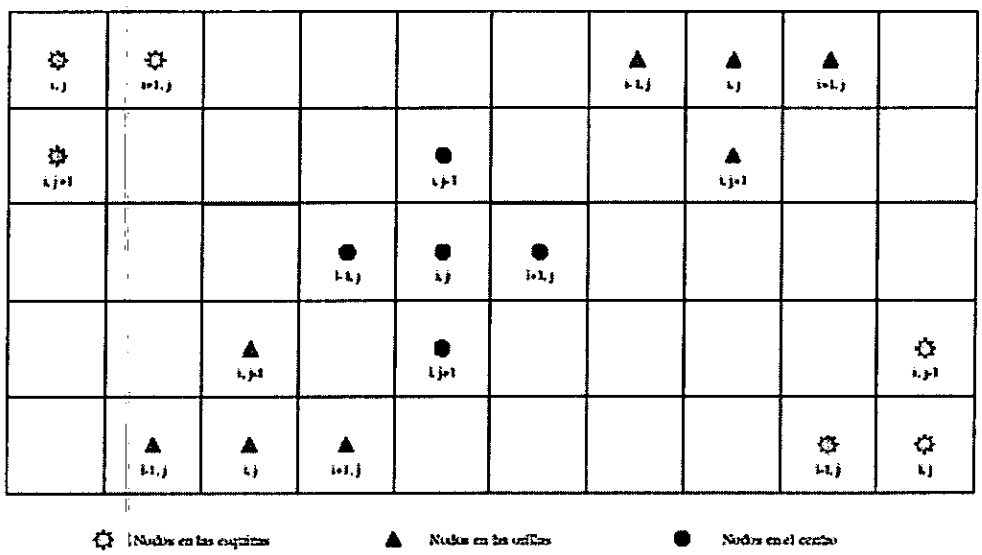


Figura 3.4. MALLA USADA EN EL ESQUEMA IMPLÍCITO, AL TIEMPO N+1.

Sustituyendo las expresiones (3.30), (3.31) y (3.32) en la (3.24), despejando $p_{i,j}^{n}$ y factorizando, se encuentra la ecuación siguiente:

$$\frac{-\beta_2}{\Delta t} p_{i,j}^{n} = \frac{k_x}{(\Delta x)^2}\left(p_{i-1,j}^{n+1} + p_{i+1,j}^{n+1}\right) - \left[2\left(\frac{k_x}{(\Delta x)^2} + \frac{k_y}{(\Delta y)^2}\right)\frac{\beta_2}{\Delta t}\right] p_{i,j}^{n+1} + \frac{k_y}{(\Delta y^2)}\left(p_{i,j-1}^{n+1} + p_{i,j+1}^{n+1}\right) + \beta_1 \quad , \tag{3.33}$$

$$i = 1, 2,.....,L \; ; \quad j = 1, 2,.....,M \; ; \quad n = 0, 1,.....,t$$

suponiendo que:

$$\psi_1 = \frac{k_x}{(\Delta x)^2} \quad , \tag{3.34}$$

$$\psi_2 = \frac{k_y}{(\Delta y)^2} \quad , \tag{3.35}$$

$$\psi_3 = \left[2\left(\frac{k_x}{(\Delta x)^2} + \frac{k_y}{(\Delta y)^2} \right) \frac{\beta_2}{\Delta t} \right] \quad . \tag{3.36}$$

Entonces, se puede conformar un sistema de ecuaciones, cuya matriz de coeficientes es penta-diagonal:

$$\begin{bmatrix} -\psi_3 & \psi_2 & 0 & \psi_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \psi_2 & -\psi_3 & \psi_2 & 0 & \psi_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \psi_2 & -\psi_3 & \psi_2 & 0 & \psi_1 & 0 & 0 & 0 & 0 \\ \psi_1 & 0 & \psi_2 & -\psi_3 & \psi_2 & 0 & \psi_1 & 0 & 0 & 0 \\ 0 & \psi_1 & 0 & \psi_2 & -\psi_3 & \psi_2 & 0 & \psi_1 & 0 & 0 \\ 0 & 0 & \psi_1 & 0 & \psi_2 & -\psi_3 & \psi_2 & 0 & \psi_1 & 0 \\ 0 & 0 & 0 & \psi_1 & 0 & \psi_2 & -\psi_3 & \psi_2 & 0 & \psi_1 \\ 0 & 0 & 0 & 0 & \psi_1 & 0 & \psi_2 & -\psi_3 & \psi_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \psi_1 & 0 & \psi_2 & -\psi_3 & \psi_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & \psi_1 & 0 & \psi_2 & -\psi_3 \end{bmatrix} \begin{bmatrix} p_{1,1}^{n+1} \\ p_{1,2}^{n+1} \\ p_{1,3}^{n+1} \\ \vdots \\ \vdots \\ p_{L,M-1}^{n+1} \\ p_{L,M}^{n+1} \end{bmatrix} = \begin{bmatrix} \frac{-\beta_2}{\Delta t} p_{1,1}^{n} - \beta_1 \\ \frac{-\beta_2}{\Delta t} p_{1,2}^{n} - \beta_1 \\ \frac{-\beta_2}{\Delta t} p_{1,3}^{n} - \beta_1 \\ \vdots \\ \vdots \\ \frac{-\beta_2}{\Delta t} p_{L,M-1}^{n} - \beta_1 \\ \frac{-\beta_2}{\Delta t} p_{L,M}^{n} - \beta_1 \end{bmatrix}$$

Como ya se mencionó en su oportunidad, la solución de una matriz penta-diagonal puede encontrarse mediante el uso del Algoritmo de Descomposición L-U. No se recomienda el uso de la subrutina DECOMP, puesto que se efectuarían operaciones innecesarias, debido a que se trata de una *matriz rala* (pocos elementos diferentes de cero).

El esquema implícito es incondicionalmente convergente y estable, no obstante, debe considerarse un rango sensato para el incremento de tiempo (Δt), pues para valores elevados de él, se presentan errores de truncamiento.

III.2.5.5 ESQUEMA DE CRANK-NICHOLSON.

El esquema de Crank-Nicholson calcula las diferenciales de las direcciones (x, y) al tiempo *"n+ ½"* ($t_{n+(½)}$). Gráficamente se representa en la Figura 3.5. Es incondicionalmente estable y convergente, además, el ciclo de cálculo consta de dos etapas: una en la dirección *"x"*, empleando una ecuación implícita en *"x"* (barrido en x), y otra en la dirección *"y"*, recurriéndose a una expresión implícita sólo en *"y"* (barrido en y).

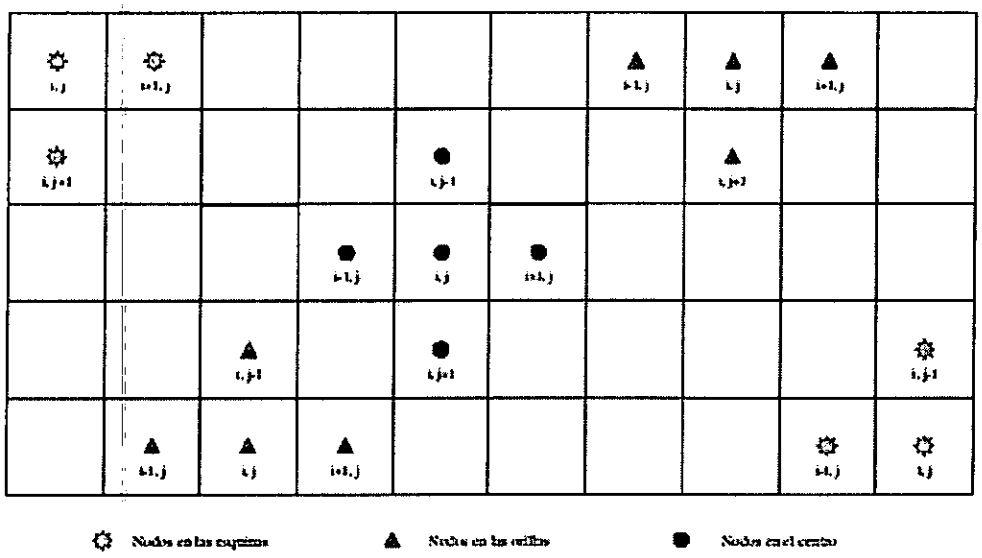


Figura 3.5. MALLA USADA EN EL ESQUEMA DE CRANK-NICHOLSON.

Para el barrido en la dirección x, se tiene:

$$\text{Diferencias finitas regresivas en tiempo}: \quad \left\{\frac{\partial p}{\partial t}\right\}_{i,j}^{n+1/2} = \frac{p_{i,j}^{n+1/2} - p_{i,j}^{n}}{\Delta t/2}, \tag{3.37}$$

$$\text{Diferencias finitas centrales en posición}: \begin{cases} \left\{\dfrac{\partial^2 p}{\partial x^2}\right\}_{i,j}^{n+1/2} = \dfrac{p_{i-1,j}^{n+1/2} - 2p_{i,j}^{n+1/2} + p_{i+1,j}^{n+1/2}}{(\Delta x)^2}, & (3.38) \\ \left\{\dfrac{\partial^2 p}{\partial y^2}\right\}_{i,j}^{n} = \dfrac{p_{i,j-1}^{n} - 2p_{i,j}^{n} + p_{i,j+1}^{n}}{(\Delta y)^2}. & (3.39) \end{cases}$$

Sustituyendo las ecuaciones (3.37), (3.38) y (3.39) en la (3.24), factorizando y despejando las $p^{n+1/2}$:

$$\frac{k_x}{(\Delta x)^2}\left(p_{i-1,j}^{n+1/2}+p_{i+1,j}^{n+1/2}\right)-\left[2\left(\frac{k_x}{(\Delta x)^2}+\frac{\beta_2}{\Delta t}\right)\right]p_{i,j}^{n+1/2}+\beta_1=-\frac{k_y}{(\Delta y)^2}\left(p_{i,j-1}^{n}+p_{i,j+1}^{n}\right)-\left[2\left(\frac{k_y}{(\Delta y)^2}+\frac{\beta_2}{\Delta t}\right)\right]p_{i,j}^{n} \quad (3.40)$$

$$i = 1, 2,.....,L ; \quad j = 1, 2,.....,M ; \quad n = 0, 1,.....,t$$

Ahora para el barrido en la dirección *"y"*:

$$\text{Diferencias finitas regresivas en tiempo: } \left\{\frac{\partial p}{\partial t}\right\}_{i,j}^{n+1}=\frac{p_{i,j}^{n+1}-p_{i,j}^{n+1/2}}{\Delta t/2}, \quad (3.41)$$

$$\text{Diferencias finitas centrales en posición: } \begin{cases}\left\{\dfrac{\partial^2 p}{\partial x^2}\right\}_{i,j}^{n+1/2}=\dfrac{p_{i-1,j}^{n+1/2}-2p_{i,j}^{n+1/2}+p_{i+1,j}^{n+1/2}}{(\Delta x)^2}, & (3.42)\\ \left\{\dfrac{\partial^2 p}{\partial y^2}\right\}_{i,j}^{n+1}=\dfrac{p_{i,j-1}^{n+1}-2p_{i,j}^{n+1}+p_{i,j+1}^{n+1}}{(\Delta y)^2}. & (3.43)\end{cases}$$

Sustituyendo las ecuaciones anteriores en la (3.24), factorizando y despejando las p^{n+1}:

$$\frac{k_y}{(\Delta y)^2}\left(p_{i,j-1}^{n+1}+p_{i,j+1}^{n+1}\right)-\left[2\left(\frac{k_y}{(\Delta y)^2}+\frac{\beta_2}{\Delta t}\right)\right]p_{i,j}^{n+1}+\beta_1=-\frac{k_x}{(\Delta x)^2}\left(p_{i-1,j}^{n+1/2}+p_{i+1,j}^{n+1/2}\right)-\left[2\left(\frac{k_x}{(\Delta x)^2}+\frac{\beta_2}{\Delta t}\right)\right]p_{i,j}^{n+1/2} \quad (3.44)$$

$$i = 1, 2,.....,L ; \quad j = 1, 2,.....,M ; \quad n = 0, 1,.....,t$$

Entonces, un intervalo de tiempo (Δt), es decir, un ciclo de cálculo, se cubrirá al resolver la ecuación (3.40) primero y posteriormente la (3.44). Esto representa la solución de 2 sistemas de ecuaciones lineales, ambos con una *matriz de coeficientes rala*, por consiguiente, es recomendable emplear el Algoritmo de Descomposición L-U, en lugar de la subrutina DECOMP, ya que dicha subrutina realizaría operaciones innecesarias.

III.2.5.6 SIMULACIÓN NUMÉRICA DE FLUJO BIDIRECCIONAL, MONOFÁSICO, TRANSITORIO, A TRAVÉS DE MEDIOS POROSOS.

Suponga, un yacimiento como el ilustrado en la Figura 3.6, se utiliza una malla relativamente simple de 6 celdas en la dirección *"x"* y 6 en la *"y"*. La frontera del yacimiento se aproxima por la línea perimetral de la misma figura. La localización de las celdas en la malla se representa en forma matricial, es decir, el índice *"i"* para los renglones y *"j"* para las columnas. Los pozos se suponen ubicados al centro de cada celda, por lo que, el pozo indicado en el esquema tiene las coordenadas (3,3). A los pozos productores se les asigna, por convención, un gasto con signo negativo y a los inyectores signo positivo.

Para este fin, se empleará el Esquema Implícito, cuya solución se basa en la aplicación de la expresión (3.33) en todos los puntos (nodos), donde la presión es desconocida. Lo anterior conformará un sistema de ecuaciones como el expuesto en el sub-tema III.2.5.4. Debido al tipo de matriz que se genera (*matriz rala*), esta se resuelve aplicando el Algoritmo de Descomposición L-U para una matriz penta-diagonal, dado el carácter de la matriz de coeficientes (consultar sección III.2.3),

Para obtener la solución del sistema de ecuaciones diferencial implicado, es necesario introducir condiciones iniciales y de frontera, éstas son:

a) Inicialmente el yacimiento tiene una presión uniforme, $P(x,y,0) = P_i$

b) No hay flujo a través de su frontera.

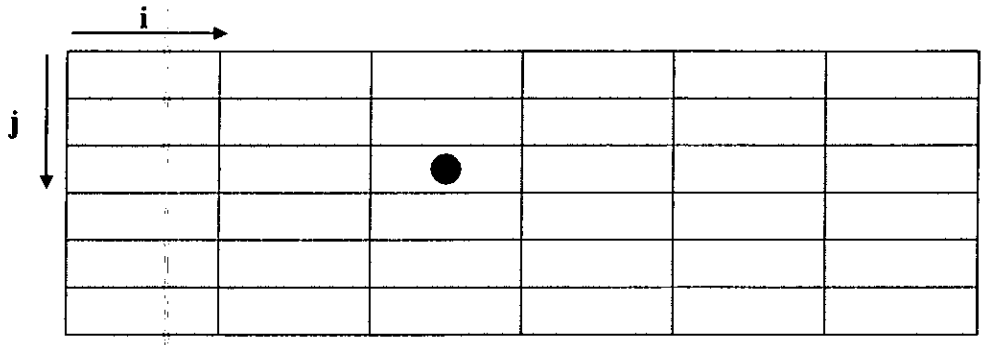


Figura 3.6. EJEMPLO DE MALLA DE UN YACIMIENTO HIPOTÉTICO.

Se puede contemplar el caso en el que la presión inicial no sea uniforme, y el gasto y la presión sean variables en los pozos. La condición dada en el inciso (b), se simula haciendo cero la permeabilidad (o en general la capacidad de flujo kh) entre dos nodos, cuando por lo menos uno de ellos esté fuera del yacimiento.

El programa de cómputo que se presenta, admite 4 pozos, sean éstos productores o inyectores. Un gasto nulo (cero) es indicativo de que el pozo no existe o está cerrado.

Se tendrán NcX • NcY ecuaciones con NcX • NcY incógnitas, siendo NcX el número de nodos en la dirección "x" y NcY el número de nodos en la "y". Un ciclo de cálculo se completará cuando se solucione el sistema de ecuaciones para un intervalo de tiempo (Δt), y la simulación llega a su fin cuando el número de ciclos cubra el tiempo de simulación supuesto.

En cada ciclo de cálculo se verifica que la presión de burbujeo (p_b) no se halla alcanzado en aquellas celdas en las cuales se localice un pozo productor, en caso afirmativo, se simula el cierre del mismo haciendo el gasto correspondiente, igual a cero. Debe señalarse que abajo de la p_b, la simulación deja de ser válida, dado que la ecuación diferencial usada es para flujo monofásico.

El diagrama de flujo se muestra, de manera esquemática y condensada, en la página 58. El programa de cómputo está disponible en el CD de consulta y tiene por nombre Simul.exe (consultar Apéndice A).

PROBLEMA.

Considerando los siguientes datos:

$k_x = k_y = 9$ [mD],	$\Delta x = \Delta y = 200$[m],	$p_i = 250$ [kg/cm^2 abs],
$\mu = 1.2$ [cp],	$\Delta t = 1$ [día],	$p_b = 240$ [kg/cm^2 abs],
$\phi = 0.25$,	$h = 47$ [m],	$C_t = 0.00015$ [cm^2/kg].

Además, se supone un pozo con un gasto de 55.3 [m^3/día], cuyas coordenadas son (3,3). Por motivos prácticos, los datos se almacenan en un archivo de acceso directo (binario). Debe recordarse que el simulador se diseñó para 6 celdas en *"x"* y 6 en *"y"*.

Cabe mencionar que se trata de un yacimiento cerrado o sea presión constante en toda la frontera.

Determínese el tiempo en que el medio poroso, aledaño al pozo productor, llegará a la p_b.

RESULTADOS.

El tiempo que deberá transcurrir para que se alcance la p_b, contemplando un ritmo de extracción constante, es de 51 días. Este resultado fue obtenido en una computadora digital, para un tiempo de simulación de 51 días. La distribución de la presión a este tiempo; se indica en la Tabla 3.3. Es importante hacer notar que si el intervalo de tiempo (Δt) se incrementa, disminuye la aproximación de la solución, y que a partir de un cierto valor de Δt, dependiendo del tamaño de las celdas de la malla, las características del yacimiento, etcétera, se presenta oscilación en los resultados.

249.80	249.54	249.16	249.55	249.84	249.95
249.54	248.75	247.13	248.77	249.63	249.89
249.16	247.13	**239.49**	247.15	249.31	249.82
249.55	248.77	247.15	248.78	249.63	249.90
249.84	249.63	249.31	249.63	249.87	249.96
249.95	249.89	249.82	249.90	249.96	249.99

Tabla 3.3. DISTRIBUCIÓN DE LA PRESIÓN (KG/CM^2) A 51 DÍAS DE SIMULACIÓN.

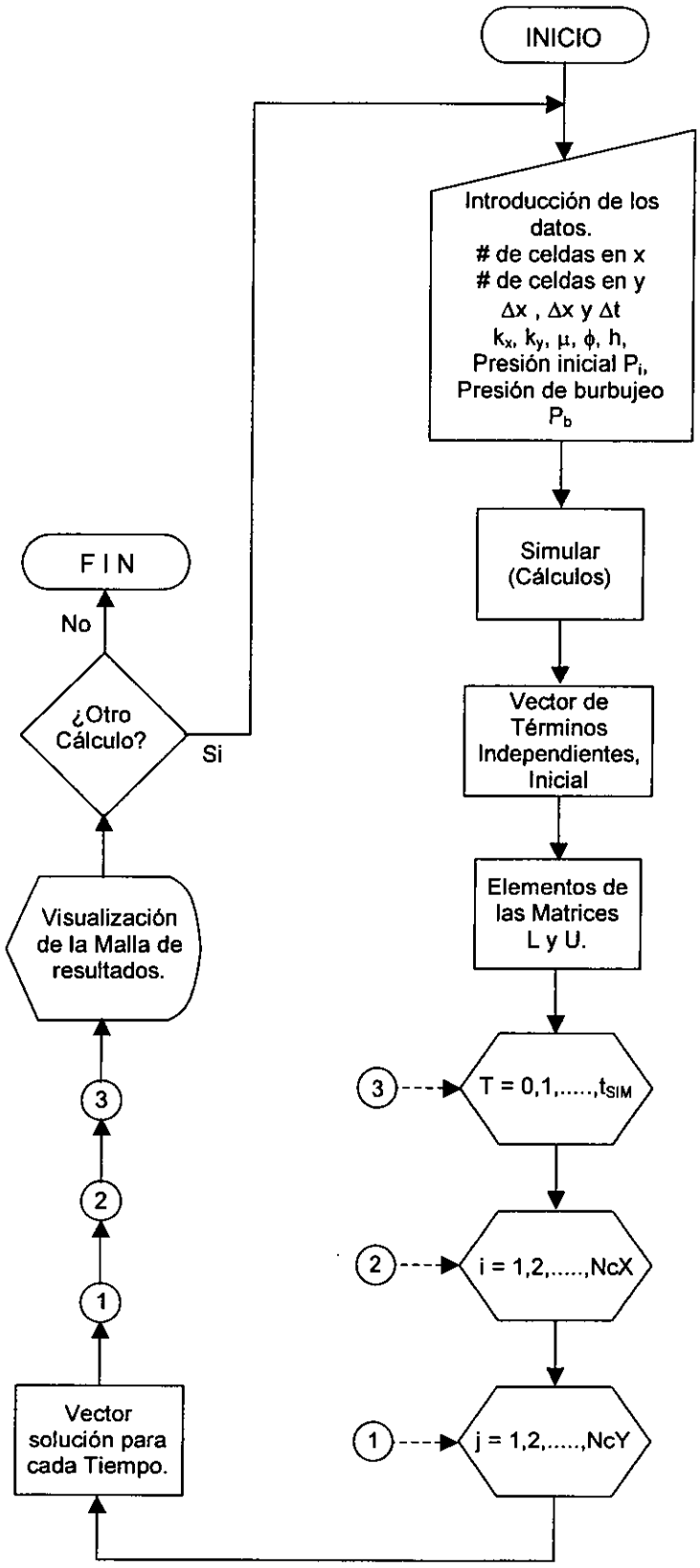


Figura 3.7. DIAGRAMA DE BLOQUES DEL SIMULADOR BIDIRECCIONAL DE FLUJO EN MEDIOS POROSOS.

III.3 SISTEMAS DE ECUACIONES NO LINEALES.

Las ecuaciones no lineales, a diferencia de las lineales, no cumplen las siguientes *Condiciones de Linealidad*[5]:

Sean V y W dos espacios vectoriales sobre un campo K. Una transformación T:V→W es lineal si para todo $\bar{v}_1, \bar{v}_2 \in V$ y para todo escalar $\alpha \in K$, se verifica:

$$1.-\ T(\bar{v}_1 + \bar{v}_2) = T(\bar{v}_1) + T(\bar{v}_2) \quad \text{(propiedad de superposición)}$$
$$2.-\ T(\alpha \bar{v}_1) = \alpha\, T(\bar{v}_1) \quad \text{(propiedad de homogeneidad)}$$

Los sistemas de ecuaciones no lineales están constituidos por funciones (circulares directas e inversas, logarítmicas, exponenciales, de potenciación y radicación), que involucran implícitamente a mas de una variable o incógnita, por lo cual, no es posible su solución mediante un "simple despeje" (solución directa), sino que deben emplearse métodos iterativos para su cálculo.

Matemáticamente, tenemos:

$$\begin{aligned} f_1(x_1, x_2, ..., x_n) &= 0 \\ f_2(x_1, x_2, ..., x_n) &= 0 \\ \vdots \quad & \quad \vdots \quad \quad \vdots \\ f_n(x_1, x_2, ..., x_n) &= 0 \quad . \end{aligned} \tag{3.45}$$

El sistema anterior involucra *"n"* funciones reales, con *"n"* variables (x_1, x_2,....,x_n), también reales, y se puede representar como:

$$\bar{x} = [x_1, x_2, ..., x_n]^T \ , \tag{3.46}$$
$$f_i(\bar{x}) = f_i(x_1, x_2, ..., x_n) \ ; \quad i = 1, 2, ..., n \ . \tag{3.47}$$

Entonces, si la solución del sistema (3.45), es la siguiente:

$$\bar{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_n]^T \quad , \tag{3.48}$$

se cumple que:

$$f_i(\bar{\alpha}) = 0 \quad ; \quad i = 1, 2, ..., n \, . \tag{3.49}$$

Por otro lado si consideramos las *"n"* funciones F(x), de tal manera que:

$$\bar{x}_i = \bar{F}_i(x) \quad ; \quad i = 1, 2, ..., n \quad , \tag{3.50}$$

lo cual implica:

$$f_j(\bar{x}) = 0 \quad , \quad j = 1, 2, ..., n \quad . \tag{3.51}$$

Tenemos, *"n"* ecuaciones (3.50) que constituyen un arreglo *"más conveniente"*, para nuestros fines, que el sistema original (3.45). En particular sea:

$$\alpha_i = F_i(\bar{\alpha}) \quad , \quad i = 1, 2, ..., n \quad . \tag{3.52}$$

Tomando en cuenta, que una primera aproximación del vector $\bar{\alpha}$, es el vector inicial mostrado a continuación:

$$\bar{x}_0 = [x_0^1, x_0^2, ..., x_0^n]^T \, . \tag{3.53}$$

Las aproximaciones sucesivas pueden definirse entonces, como:

$$\bar{x}_k = [x_k^1, x_k^2, ..., x_k^n]^T \quad ; \quad k = 1, 2, ... \quad . \tag{3.54}$$

Cada elemento del vector (3.54) se obtiene a partir de la siguiente ecuación de recurrencia:

$$x_k^i = F_i\left(x_{k-1}^1, x_{k-1}^2, ..., x_{k-1}^n\right) = F_i\left(\bar{x}_{k-1}\right). \tag{3.55}$$

Suponga que existe una región $\Re$, donde:

$$\left|x_j - \alpha_j\right| \leq \textit{valor moderamente grande} \quad , \quad j = 1, 2, ..., n$$

Y para todo $\bar{x}$ en $\Re$, existe un número positivo $\mu < 1$, tal que:

$$\sum_{j=1}^{n} \left| \frac{\partial F_i(\bar{x})}{\partial x_j} \right| \leq \mu \ , \tag{3.56}$$

por lo tanto, si el vector inicial $\bar{x}_0$ pertenece al conjunto $\Re$, el método iterativo expresado por la ecuación (3.54), convergerá a la solución del sistema, es decir:

$$\lim_{k \to \infty} \bar{x}_k = \bar{\alpha} \ . \tag{3.57}$$

A diferencia de los métodos para sistemas de ecuaciones lineales, denominados *"exactos"*, los concernientes a sistemas no lineales, acarrean errores de redondeo de un paso a otro. Esto, debido a que el resultado en cada iteración puede ser visualizado como una nueva suposición o como una aproximación al vector solución. Pueden permitirse errores substanciales, con tal que no agiganten el error final o no sobrepasen la tolerancia establecida.

Aunque existe una gran cantidad de métodos para la solución de sistemas de ecuaciones no lineales, se presentan sólo los más comunes: Jacobi, Gauss-Seidel, Newton-Raphson y Newton-Rapshon Mejorado siendo este último la mejor alternativa.

III.3.1 MÉTODO ITERATIVO DE JACOBI.

Considérese un sistema no lineal cualquiera, como el sistema (3.45), en el cual, el vector solución es:

$$\bar{x} = [x_1, x_2, ..., x_n]^T \ , \tag{3.58}$$

y:

$$f_i(\bar{x}) = f_i(x_1, x_2, ..., x_n) \quad , \quad i = 1, 2, ..., n \ . \tag{3.59}$$

Este método consiste en transformar el sistema original, en uno del tipo de la ecuación (3.50). Cuya solución se logrará mediante un ciclo iterativo, apoyándose en la ecuación de recurrencia (3.55), la cual es:

$$x_k^i = F_i\left(x_{k-1}^1, x_{k-1}^2, ..., x_{k-1}^n\right) = F_i\left(\bar{x}_{k-1}\right) \ , \tag{3.60}$$

donde:

k = número de iteración , k = 1, 2,... .

Además, se contempla una tolerancia y un vector inicial como el siguiente:

$$\bar{x}_0 = \left[x_0^1, x_0^2, ..., x_0^n\right]^T \tag{3.61}$$

El grado de aproximación en la solución, casi siempre, puede ser mejorado al incrementar el número de iteraciones y depurar el proceso de cálculo en cada una de ellas. Esta técnica es de lenta convergencia y emplea memoria RAM para los elementos x_i^{k+1} y x_i^k, en cada iteración.

III.3.2 MÉTODO ITERATIVO DE GAUSS-SEIDEL.

Tomando en cuenta el sistema de ecuaciones (3.45), el cual tiene como vector solución a:

$$\bar{x} = [x_1, x_2, ..., x_n]^T \quad , \tag{3.62}$$

y:

$$f_i(\bar{x}) = f_i(x_1, x_2, ..., x_n) \quad ; \quad i = 1, 2, ..., n \quad . \tag{3.63}$$

Entonces:

$$x_i = F_i(\bar{x}) \quad ; \quad i = 1, 2, ..., n \quad . \tag{3.64}$$

La solución se basa en la utilización de la siguiente ecuación de recurrencia:

$$x_k^i = F_i\left(x_k^1, x_k^2, ..., x_k^{i-1}, x_{k-1}^i, x_{k-1}^n\right) \quad , \tag{3.65}$$

donde:

k = número de iteración , k = 1, 2,... .

Se considera también, una tolerancia y un vector inicial, el cual puede ser:

$$\bar{x}_0 = [x_0^1, x_0^2, ..., x_0^n]^T \quad . \tag{3.66}$$

Este método converge más rápido que el de Jacobi, ya que una vez calculado el componente x_{k+1}^i, lo aplica inmediatamente en la misma iteración.

III.3.3 MÉTODO ITERATIVO DE NEWTON-RAPHSON.

El algoritmo para resolver un sistema de ecuaciones como el (3.45), usando el método de Newton-Raphson, puede resumirse así:

1) Elegir un vector inicial, esto es:

$$\bar{x}_k = \bar{x}_0 = \left[x_0^1, x_0^2, ..., x_0^n\right]^T \quad . \tag{3.67}$$

2) Resolver el sistema de ecuaciones que se muestra a continuación:

$$F\left(\bar{x}_k\right)\bar{\Delta}_k = -\bar{f}\left(\bar{x}_k\right) \quad , \tag{3.68}$$

donde:

$$F_{ij}\left(\bar{x}_k\right) = \frac{\partial f_i}{\partial x_j}\left(\bar{x}_k\right) \quad , \tag{3.69}$$

$$\bar{f}\left(\bar{x}_k\right) = \left[f_1\left(\bar{x}_k\right), f_2\left(\bar{x}_k\right), ..., f_n\left(\bar{x}_k\right)\right]^T \quad , \tag{3.70}$$

$$\bar{\Delta}_k = \left[\Delta_{1_k}, \Delta_{2_k}, ..., \Delta_{n_k}\right]^T \quad . \tag{3.71}$$

i = 1, 2,.....,n ; j = 1, 2,.....,n ; k = número de iteración (1, 2,...) .

El sistema anterior, como se observa, es lineal. El vector $\bar{\Delta}_k$ es la solución del sistema (también es denominado vector de incrementos) y la matriz $F\left(\bar{x}_k\right)$ es el Jacobiano.

3) Calcular la aproximación de la solución del sistema de ecuaciones no lineales original (ecuación 3.45), o sea:

$$\bar{x}_{k+1} = \bar{x}_k + \bar{\Delta}_k \quad . \tag{3.72}$$

4) Verificar la posible convergencia de la aproximación de la solución del sistema de ecuaciones no lineales. Se sugiere la siguiente prueba:

$$\left|x_{k+1}^{i} - x_{k}^{i}\right| \leq \text{tolerancia} \quad , \quad i = 1, 2, ..., n . \tag{3.73}$$

Si la expresión anterior se cumple para toda *"i"*, entonces x_{k+1}^{i} será la aproximación de la solución del sistema de ecuaciones no lineales $(\bar{\alpha})$ (ver Figura 3.8), finalizando el cálculo. Pero si la prueba falla para alguna *"i"*, el proceso se repite a partir del paso 2, y el cálculo iterativo continuará hasta verificarse el criterio de tolerancia o hasta que se exceda algún límite de iteraciones fijado $(k_{máx})$.

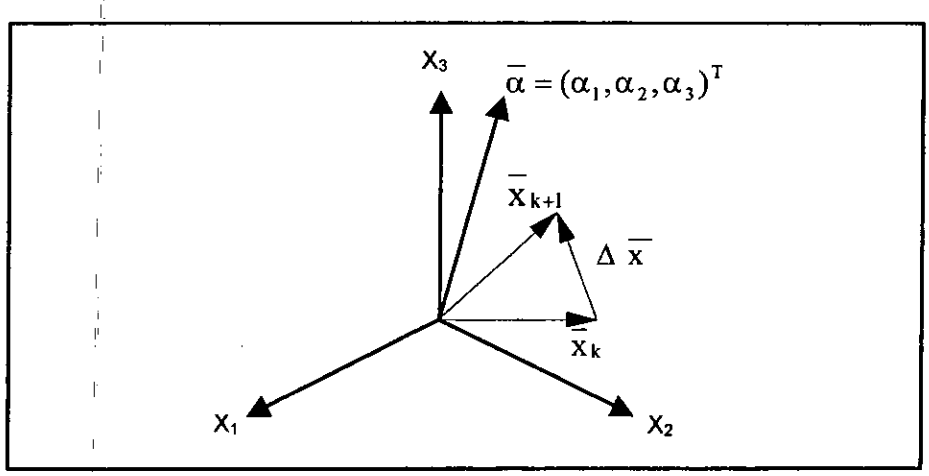


Figura 3.8. MÉTODO DE NEWTON RAPHSON.
Suponiendo un Sistema de 3 Ecuaciones con 3 Incógnitas.

Si los componentes de $F(\bar{x})$ son continuos en la vecindad de un punto definido por el vector $\bar{\alpha}$, tal que:

$$\bar{f}(\bar{\alpha}) = 0 . \tag{3.74}$$

Y si el determinante de $f(\bar{\alpha}) \neq 0$ y $\bar{x}_0$ está muy cercano $\bar{\alpha}$, se asegura la convergencia del método[2], es decir:

$$\lim_{k \to \infty} \bar{x}_k = \bar{\alpha} \quad . \tag{3.75}$$

Debido a que se usan lo últimos valores x_{i-1}^{k+1} para calcular x_i^{k+1}, el requerimiento de memoria RAM es menor que en el método iterativo de Jacobi, además, su convergencia es mucho más rápida.

III.3.4 MÉTODO ITERATIVO DE NEWTON-RAPHSON MEJORADO[20].

La modificación de este método implica la utilización de la fórmula de Taylor de segundo orden y del Hessiano de las 2as derivadas para resolver el sistema de ecuaciones de la forma $A\bar{x} = \bar{b}$.

La Fórmula de Taylor en forma general puede expresarse como:

$$F_i(\bar{x}+\Delta\bar{x}) = F_i(\bar{x}) + J_i(\bar{x})\Delta\bar{x} + \frac{1}{2}\Delta\bar{x}H_i(\bar{x})\Delta\bar{x}^T + r_i(\Delta\bar{x}) \;\; ; \;\; i = 1,2,3,\ldots,n \;, \qquad (3.76)$$

donde:

F_i : función de residuos.

J_i :Jacobiano de la función.

H_i : matriz Hessiana que contiene las segundas derivadas.

r_i : función residuo

$$J_i(\bar{x}) = \left[\frac{\partial F_i}{\partial x_1} \quad \frac{\partial F_i}{\partial x_2} \quad \frac{\partial F_i}{\partial x_3} \quad \cdots \quad \frac{\partial F_i}{\partial x_n}\right], \qquad (3.77)$$

$$H_i(\bar{x}) = \begin{bmatrix} \frac{\partial^2 F_i}{\partial x_1^2} & \frac{\partial^2 F_i}{\partial x_1 \partial x_2} & \frac{\partial^2 F_i}{\partial x_1 \partial x_3} & \cdots & \frac{\partial^2 F_i}{\partial x_1 \partial x_n} \\ \frac{\partial^2 F_i}{\partial x_2 \partial x_1} & \frac{\partial^2 F_i}{\partial x_2^2} & \frac{\partial^2 F_i}{\partial x_2 \partial x_3} & \cdots & \frac{\partial^2 F_i}{\partial x_2 \partial x_n} \\ \frac{\partial^2 F_i}{\partial x_3 \partial x_1} & \frac{\partial^2 F_i}{\partial x_3 \partial x_2} & \frac{\partial^2 F_i}{\partial x_3^2} & \cdots & \frac{\partial^2 F_i}{\partial x_3 \partial x_n} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \frac{\partial^2 F_i}{\partial x_n \partial x_1} & \frac{\partial^2 F_i}{\partial x_n \partial x_2} & \frac{\partial^2 F_i}{\partial x_n \partial x_3} & \cdots & \frac{\partial^2 F_i}{\partial x_n^2} \end{bmatrix}, \qquad (3.78)$$

$$r(\Delta x) = r(x_1, x_2)\,. \tag{3.79}$$

$$\Delta\bar{x} = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_n \end{bmatrix} \quad ; \quad \Delta x = x_{1_{i+1}} - x_{1_i} \quad .$$

Por la definición del teorema de Schwarz de derivadas parciales tenemos que:

$$\frac{\partial^2 F_i}{\partial x_1 \partial x_2} = \frac{\partial^2 F_i}{\partial x_2 \partial x_1}\,, \tag{3.80}$$

entonces la matriz Hessiana es simétrica.

El método de Newton-Raphson puede ser modificado tomando los cálculos de las segundas derivadas. Esto es, si las estimaciones de x_0, y_0 son incrementadas respectivamente por x_1, x_2, entonces las aproximaciones de segundo orden del cambio en los resultados en $F_i(x)$ esta dada por la expresión siguiente.

$$F_i = \frac{\partial F_i}{\partial x_1}\delta x_1 + \frac{\partial F_i}{\partial x_2}\delta x_2 + \frac{1}{2}\left[\frac{\partial^2 F_i}{\partial x_1^2}\delta x_1^2 + 2\frac{\partial^2 F_i}{\partial x_1 \partial x_2}\delta x_1 \delta x_2 + \frac{\partial^2 F_i}{\partial x_2^2}\delta x_2^2\right] \quad ; \quad i = 1,2,3,\ldots,n\,, \tag{3.81}$$

agrupando simplificando:

$$F_i = \left[\frac{\partial F_i}{\partial x_1} + \frac{1}{2}\frac{\partial^2 F_i}{\partial x_1^2}\delta x_1 + \frac{1}{2}\frac{\partial^2 F_i}{\partial x_1 \partial x_2}\delta x_2\right]\delta x_1 + \left[\frac{\partial F_i}{\partial x_2} + \frac{1}{2}\frac{\partial^2 F_i}{\partial x_2^2}\delta x_2 + \frac{1}{2}\frac{\partial^2 F_i}{\partial x_1 \partial x_2}\delta x_1\right]\delta x_2 \quad ; \quad i = 1,2,3,\ldots,n\,. \tag{3.82}$$

Con estas ecuaciones se forma un sistema matricial del tipo $A\bar{x} = -\bar{F}$.

Como se puede observar en la ecuación anterior, si se aumenta el número de incógnitas la ecuación será mucho más compleja y debido a esto, sólo se tomaran dos incógnitas.

Para evaluar F(x), J($\bar{x}$) y el H($\bar{x}$) se usan los valores iniciales para la primera iteración, estos para la segunda iteración y así sucesivamente.

Como en A se usan valores conocidos, requiere aproximar las δx_1 y δx_2 que se encuentran dentro de los paréntesis rectangulares, para esto se usa el artificio[20] siguiente

$$\delta x_i = -\frac{F_i}{\frac{\partial F_i}{\partial x_i}} \quad ; \quad i = 1, 2, 3, \ldots, n \ , \tag{3.83}$$

aplicando lo anterior se llega a:

$$F_i = \left[\frac{\partial F_i}{\partial x_1} - \frac{1}{2} \frac{F_i}{\frac{\partial F_i}{\partial x_1}} \frac{\partial^2 F_i}{\partial x_1^2} - \frac{1}{2} \frac{F_i}{\frac{\partial F_i}{\partial x_2}} \frac{\partial^2 F_i}{\partial x_1 \partial x_2} \right] \delta x_1 + \left[\frac{\partial F_i}{\partial x_2} - \frac{1}{2} \frac{F_i}{\frac{\partial F_i}{\partial x_2}} \frac{\partial^2 F_i}{\partial x_2^2} - \frac{1}{2} \frac{F_i}{\frac{\partial F_i}{\partial x_1}} \frac{\partial^2 F_i}{\partial x_1 \partial x_2} \right] \delta x_2 \ ; \ i = 1, 2, 3, \ldots, n \ , \tag{3.84}$$

que es la fórmula de Taylor de segundo orden involucrando al Hessiano para su solución.

Finalmente aplicando la forma de $A\bar{x} = -\bar{F}$ a la ecuación anterior se tiene que:

$$\begin{bmatrix} \frac{\partial F_1}{\partial x_1} - \frac{1}{2} \frac{F_1}{\frac{\partial F_1}{\partial x_1}} \frac{\partial^2 F_1}{\partial x_1^2} - \frac{1}{2} \frac{F_1}{\frac{\partial F_1}{\partial x_2}} \frac{\partial^2 F_1}{\partial x_1 \partial x_2} & \frac{\partial F_1}{\partial x_2} - \frac{1}{2} \frac{F_1}{\frac{\partial F_1}{\partial x_2}} \frac{\partial^2 F_1}{\partial x_2^2} - \frac{1}{2} \frac{F_1}{\frac{\partial F_1}{\partial x_1}} \frac{\partial^2 F_1}{\partial x_1 \partial x_2} \\ \frac{\partial F_2}{\partial x_1} - \frac{1}{2} \frac{F_2}{\frac{\partial F_2}{\partial x_1}} \frac{\partial^2 F_2}{\partial x_1^2} - \frac{1}{2} \frac{F_2}{\frac{\partial F_2}{\partial x_2}} \frac{\partial^2 F_2}{\partial x_1 \partial x_2} & \frac{\partial F_2}{\partial x_2} - \frac{1}{2} \frac{F_2}{\frac{\partial F_2}{\partial x_2}} \frac{\partial^2 F_2}{\partial x_2^2} - \frac{1}{2} \frac{F_2}{\frac{\partial F_2}{\partial x_1}} \frac{\partial^2 F_2}{\partial x_1 \partial x_2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -F_1 \\ -F_2 \end{bmatrix}, \tag{3.85}$$

que es el sistema final a resolver, con los métodos para resolver Sistemas de Ecuaciones Lineales.

Como se puede observar, si se desean hacer todos los cálculos a mano sería muy complicado y tardado, y más cuando se aumenta el número de las incógnitas se vuelve aún más complejo debido a que va creciendo la expresión.

Como siguiente paso se procederá a dar solución al sistema de ecuaciones realizando una serie de iteraciones partiendo de valores iniciales arbitrarios (x_1, x_2), cabe mencionar que mientras más alejados estén estos valores de la solución mayor será el número de iteraciones que se tendrán que hacer.

Una vez obtenidos los valores de δx se les comparará con una tolerancia (Tol) que se establecerá según se crea conveniente para el problema a resolver, para así saber si ya se llegó al resultado o se continuará con un nuevo procedimiento iterativo, como se muestra a continuación:

$$|\delta x_1| > \text{Tol} \text{ ó } |\delta x_2| > \text{Tol} . \tag{3.86}$$

Si lo anterior es verdad se hará un nuevo cálculo, pero ahora los nuevos valores serán calculados como sigue:

$$x_{1\,\text{inicial}} + \delta x_{1\,\text{calculado}} = x_{1\,\text{nuevo}} , \tag{3.87}$$

$$x_{2\,\text{inicial}} + \delta x_{2\,\text{calculado}} = x_{2\,\text{nuevo}} . \tag{3.88}$$

Con estos nuevos valores se procederá a realizar una nueva iteración y así sucesivamente hasta llegar al resultado deseado del sistema de ecuaciones.

A continuación se propone un ejemplo numérico de un sistema de ecuaciones a resolver por los métodos de Newton-Raphson y Newton-Raphson Mejorado, también se mostrará el código y pantalla de resultados que se programó para resolver el ejemplo y que fue desarrollado en Visual Basic 6.0®.

1. MÉTODO DE NEWTON-RAPHSON.

Sea el siguiente sistema de ecuaciones:

$$F_1 = x^2 + y^2 - 4 = 0 \quad , \tag{3.89}$$

$$F_2 = \frac{x^2}{9} + y^2 - 1 = 0 \quad . \tag{3.90}$$

Determinar su solución partiendo de los siguientes valores iniciales y tomando una tolerancia de 0.001 (Tol = 0.001):

$$x_1 = 1, x_2 = 1 \quad \text{o} \quad \text{de otra forma} \quad x_0 = 1, y_0 = 1 .$$

Solución:

Aplicando la forma $A\bar{x} = -\bar{F}$, el sistema anterior quedará de la siguiente manera:

$$\begin{bmatrix} \frac{\partial F_1}{\partial x} & \frac{\partial F_1}{\partial y} \\ \frac{\partial F_2}{\partial x} & \frac{\partial F_2}{\partial y} \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} = \begin{bmatrix} -F_1 \\ -F_2 \end{bmatrix} . \tag{3.91}$$

En cada iteración, se calculará la δx y δy para resolver el sistema.

Donde se continuará iterando aplicando las siguientes expresiones:

$$x_0 + \delta x = \text{Que será el nuevo valor de } x_0 \quad , \quad y_0 + \delta y = \text{Que será el nuevo valor de } y_0 .$$

Derivando las ecuaciones del sistema se tiene que:

$$\frac{\partial F_1}{\partial x} = 2x \quad , \quad \frac{\partial F_1}{\partial y} = 2y$$

$$\frac{\partial F_2}{\partial x} = \frac{2}{9}x \quad , \quad \frac{\partial F_2}{\partial y} = 2y$$

Sustituyendo valores iniciales a las ecuaciones del sistema se tiene que:

$$x_0 = 1 \, , \, y_0 = 1; \quad F_1 = -2, \quad F_2 = 1/9 \, .$$

Ahora, sustituyendo todos los valores a la expresión de la forma $A\bar{x} = -\bar{F}$ se tendrá lo siguiente:

$$\begin{bmatrix} 2 & 2 \\ \frac{2}{9} & 2 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} = \begin{bmatrix} -(-2) \\ -\frac{1}{9} \end{bmatrix} . \qquad (3.92)$$

Resolviendo el sistema anterior se obtienen los nuevos valores de "x" y de "y":

$$\begin{aligned} \delta x &= 1.1875 \quad , \quad x_0 + \delta x = 2.1875 \\ \delta y &= -0.1875 \quad , \quad y_0 + \delta y = 0.8125 \end{aligned} \, ,$$

Evaluamos las funciones F_1 y F_2, como $|\delta x| > Tol$ y $|\delta y| > Tol$ se realiza otro ciclo iterativo.

$$x_0 = 2.1875 \, , \, y_o = 0.8125; \quad F_1 = 1.445, \quad F_2 = 0.192 \, .$$

$$\begin{bmatrix} 4.375 & 1.625 \\ 0.486 & 1.625 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} = \begin{bmatrix} -1.445 \\ -0.192 \end{bmatrix} . \qquad (3.93)$$

Se obtienen los nuevos valores:

$$\begin{aligned} \delta x &= -0.322 \quad , \quad x_0 + \delta x = 1.8655 \\ \delta y &= -0.0218 \quad , \quad y_0 + \delta y = 0.7907 \end{aligned} \quad ,$$

se procederá a hacer otra iteración de la misma manera, ya que $|\delta x| > Tol$ y $|\delta y| > Tol$.

$$x_0 = 1.8655 , y_o = 0.7907; \quad F_1 = 0.1052, \quad F_2 = 0.0119.$$

$$\begin{bmatrix} 3.7310 & 1.5818 \\ 0.4146 & 1.5818 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} = \begin{bmatrix} -0.1052 \\ -0.0119 \end{bmatrix}. \qquad (3.94)$$

Y se obtiene:

$$\begin{aligned} \delta x &= -0.0281 \quad , \quad x_0 + \delta x = 1.8374 \\ \delta y &= -0.00015 \quad , \quad y_0 + \delta y = 0.79055 \end{aligned} \quad ,$$

$$x_0 = 1.8374 , y_o = 0.79055; \quad F_1 = 0.001008, \quad F_2 = 0.000084.$$

Estos últimos valores pueden ser muy buenas aproximaciones ya que como se ve F_1 y F_2 son prácticamente iguales a cero.

2. MÉTODO DE NEWTON-RAPHSON MEJORADO[20].

Del sistema de ecuaciones (3.89) y (3.90). Determinar su solución partiendo de los mismos valores iniciales:

Solución:

Aplicando la forma $A\bar{x} = -\bar{F}$ el sistema quedará de la manera siguiente:

$$\begin{bmatrix} \frac{\partial F_1}{\partial x} - \frac{1}{2}\frac{F_1}{\frac{\partial F_1}{\partial x}}\frac{\partial^2 F_1}{\partial x^2} - \frac{1}{2}\frac{F_1}{\frac{\partial F_1}{\partial y}}\frac{\partial^2 F_1}{\partial x \partial y} & \frac{\partial F_1}{\partial y} - \frac{1}{2}\frac{F_1}{\frac{\partial F_1}{\partial y}}\frac{\partial^2 F_1}{\partial y^2} - \frac{1}{2}\frac{F_1}{\frac{\partial F_1}{\partial x}}\frac{\partial^2 F_1}{\partial x \partial y} \\ \frac{\partial F_2}{\partial x} - \frac{1}{2}\frac{F_2}{\frac{\partial F_2}{\partial x}}\frac{\partial^2 F_2}{\partial x^2} - \frac{1}{2}\frac{F_2}{\frac{\partial F_2}{\partial y}}\frac{\partial^2 F_2}{\partial x \partial y} & \frac{\partial F_2}{\partial y} - \frac{1}{2}\frac{F_2}{\frac{\partial F_2}{\partial y}}\frac{\partial^2 F_2}{\partial y^2} - \frac{1}{2}\frac{F_2}{\frac{\partial F_2}{\partial x}}\frac{\partial^2 F_2}{\partial x \partial y} \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} = \begin{bmatrix} -F_1 \\ -F_2 \end{bmatrix}. \qquad (3.95)$$

En cada iteración, se calculará la δx y δy para resolver el sistema. Donde se continuará iterando aplicando las siguientes expresiones:

$$x_0 + \delta x = \text{Que será el nuevo valor de } x_0 \quad , \quad y_0 + \delta y = \text{Que será el nuevo valor de } y_0$$

Derivando las ecuaciones del sistema se tiene que:

$$\frac{\partial F_1}{\partial x} = 2x \quad , \quad \frac{\partial^2 F_1}{\partial x^2} = 2 \quad , \quad \frac{\partial^2 F_1}{\partial x \partial y} = 0$$

$$\frac{\partial F_1}{\partial y} = 2y \quad , \quad \frac{\partial^2 F_1}{\partial y^2} = 2$$

$$\frac{\partial F_2}{\partial x} = \frac{2}{9}x \quad , \quad \frac{\partial^2 F_2}{\partial x^2} = \frac{2}{9} \quad , \quad \frac{\partial^2 F_2}{\partial x \partial y} = 0$$

$$\frac{\partial F_2}{\partial y} = 2y \quad , \quad \frac{\partial^2 F_2}{\partial y^2} = 2$$

Sustituyendo valores iniciales $x_0 = 1$, $y_0 = 1$ en a las ecuaciones del sistema se tiene que:

$$F_1 = -2, \quad F_2 = 1/9 \ .$$

Ahora, sustituyendo todos los valores a la expresión de la forma $A\bar{x} = -\bar{F}$ se tendrá lo siguiente:

$$\begin{bmatrix} 2-\frac{1}{2}\left(\frac{-2}{2}\right)2-0 & 2-\frac{1}{2}\left(\frac{-2}{2}\right)2-0 \\ \frac{2}{9}-\frac{1}{2}\left(\frac{1/9}{2/9}\right)\frac{2}{9}-0 & 2-\frac{1}{2}\left(\frac{1/9}{2}\right)2-0 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} = \begin{bmatrix} -(-2) \\ -\frac{1}{9} \end{bmatrix} \qquad (3.96)$$

con lo cual se obtienen los nuevos valores:

$$\begin{aligned} \delta x = 0.7916 \ , \quad & x_0 + \delta x = 1.7916 \\ \delta y = -0.125 \ , \quad & y_0 + \delta y = 0.8750 \end{aligned} \ ,$$

se procede a hacer la siguiente iteración de la misma manera, ya que $|\delta x| > \text{Tol}$ y $|\delta y| > \text{Tol}$.

$$x_0 = 0.7916 \, , y_0 = 0.8750; \quad F_1 = -0.0246, \quad F_2 = 0.1222 \, .$$

$$\begin{bmatrix} 3.5832-\frac{1}{2}\left(\frac{-0.0246}{3.5832}\right)2-0 & 1.75-\frac{1}{2}\left(\frac{-0.0246}{1.75}\right)2-0 \\ 0.3981-\frac{1}{2}\left(\frac{0.1222}{0.3981}\right)\frac{2}{9}-0 & 1.75-\frac{1}{2}\left(\frac{0.1222}{1.75}\right)2-0 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} = \begin{bmatrix} -(-0.0246) \\ -0.1222 \end{bmatrix} \qquad (3.97)$$

se obtienen los nuevos valores:

$$\begin{aligned} \delta x = -0.0477 \ , \quad & x_0 + \delta x = 1.8399 \\ \delta y = -0.0831 \ , \quad & y_0 + \delta y = 0.7919 \end{aligned} \ ,$$

se procede a hacer otra iteración más, ya que $|\delta x| > Tol$ y $|\delta y| > Tol$.

$$x_0 = 1.8399\,,\ y_o = 0.7919;\quad F_1 = 0.0101,\quad F_2 = 0.0030.$$

$$\begin{bmatrix} 3.6798-\frac{1}{2}\left(\frac{0.0101}{3.6798}\right)2-0 & 1.5838-\frac{1}{2}\left(\frac{0.0101}{1.5838}\right)2-0 \\ 0.4089-\frac{1}{2}\left(\frac{0.0030}{0.4089}\right)\frac{2}{9}-0 & 1.5838-\frac{1}{2}\left(\frac{0.0030}{1.5838}\right)2-0 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} = \begin{bmatrix} -0.0101 \\ -0.0030 \end{bmatrix} \qquad (3.98)$$

se obtienen los nuevos valores:

$$\begin{aligned} \delta x &= -0.0004 \quad , \quad x_0 + \delta x = 1.8394 \\ \delta y &= -0.0020 \quad , \quad y_0 + \delta y = 0.7898 \end{aligned} \quad ,$$

$$x_0 = 1.8394\,,\ y_o = 0.7898;\quad F_1 = 0.0071,\quad F_2 = 0.0003.$$

Estos últimos valores pueden ser muy buenas aproximaciones ya que como se ve F_1 y F_2 se aproximan a cero. Para este ejemplo en particular, el método de Newton-Raphson Mejorado producen una solución que es casi igual a la producida por el método Newton Raphson, cabe mencionar que cuando se tiene mayor número de ecuaciones el numero de iteraciones se magnifica uno con respecto al otro.

Una vez realizado el ejemplo a mano por ambos métodos, se mostrara el código del programa y la pantalla de resultados también de ambos métodos.

MÉTODO NEWTON-RAPHSON.

```
tiempo1 = Timer
x0 = Val(Form3.Text1.Text) : y0 = Val(Form3.Text2.Text)
iter = 0
100
iter = iter + 1
'Evalúa la primera función y las primeras derivadas en x0 y y0
f1f = (x0 ^ 2) + (y0 ^ 2) – 4
f1fx = 2 * x0
f1fy = 2 * y0
'Evalúa la segunda función y las primeras derivadas en x0 y y0
f2f = ((x0 ^ 2) / 9) + (y0 ^ 2) - 1
f2fx = (2 / 9) * x0
f2fy = 2 * y0
'Se establece el Jacobiano
Jacobiano(1, 1) = f1fx : Jacobiano(1, 2) = f1fy
Jacobiano(2, 1) = f2fx : Jacobiano(2, 2) = f2fy
'Evalúa la primera función y las primeras derivadas en x1 y y1
f1f = (x0 ^ 2) + (y0 ^ 2) - 4
'Evalúa la segunda función y las primeras derivadas en x1 y y1
f2f = ((x0 ^ 2) / 9) + (y0 ^ 2) - 1
If Abs(f1f) <= tolerancia And Abs(f2f) <= tolerancia Then
   GoTo 200
Else
   GoTo 100
End If
200
tiempo2 = Timer
tiempo_total = tiempo2 - tiempo1
```

```
With Form3
   .Label7.Caption = Format(x0, "###0.0###")
   .Label8.Caption = Format(y0, "###0.0###")
   .Label21.Caption = Format(f1f, "###0.0###")
   .Label23.Caption = Format(f2f, "###0.0###")
   .Label9.Caption = iter
   .Label17.Caption = Format(tiempo_total, "##0.0#####")
End With
```

MÉTODO NEWTON-RAPHSON MEJORADO

```
tiempo1 = Timer
x0 = Val(Form3.Text1.Text) : y0 = Val(Form3.Text2.Text)
iter = 0
100
iter = iter + 1
'Evalúa la primera función y las primeras derivadas en x0 y y0
f1f = (x0 ^ 2) + (y0 ^ 2) - 4
f1fx = 2 * x0
f1fy = 2 * y0
f1fxy = 0
f1fxx = 2
f1fyy = 2
'Evalúa la segunda función y las primeras derivadas en x0 y y0
f2f = ((x0 ^ 2) / 9) + (y0 ^ 2) - 1
f2fx = (2 / 9) * x0
f2fy = 2 * y0
f2fxy = 0
f2fxx = 2 / 9
f2fyy = 2
```

```
'Se establece el Hessiano
Hessiano (1, 1) = f1fx - ((f1fxx * f1f) / (2 * f1fx)) - ((f1fxy * f1f) / (2 * f1fy))
Hessiano (1, 2) = f1fy - ((f1fyy * f1f) / (2 * f1fy)) - ((f1fxy * f1f) / (2 * f1fx))
Hessiano (2, 1) = f2fx - ((f2fxx * f2f) / (2 * f2fx)) - ((f2fxy * f2f) / (2 * f2fy))
Hessiano (2, 2) = f2fy - ((f2fyy * f2f) / (2 * f2fy)) - ((f2fxy * f2f) / (2 * f2fx))
Vector_b(1) = -f1f
Vector_b(2) = -f2f
x0 = x0 + Vector_b(1)
y0 = y0 + Vector_b(2)
'Evalúa la primera función y las primeras derivadas en x1 y y1
f1f = (x0 ^ 2) + (y0 ^ 2) - 4
'Evalúa la segunda función y las primeras derivadas en x1 y y1
f2f = ((x0 ^ 2) / 9) + (y0 ^ 2) - 1
If Abs(f1f) <= tolerancia And Abs(f2f) <= tolerancia Then
   GoTo 200
Else
   GoTo 100
End If
200  tiempo2 = Timer
tiempo_total = tiempo2 - tiempo1
With Form3
   .Label12.Caption = Format(x0, "###0.0###")
   .Label11.Caption = Format(y0, "###0.0###")
   .Label28.Caption = Format(f1f, "###0.0###")
   .Label26.Caption = Format(f2f, "###0.0###")
   .Label10.Caption = iter
   .Label19.Caption = Format(tiempo_total, "##0.0#####")
End With
```

En la Figura 3.9 se muestra los resultados obtenidos por ambos métodos en donde se ve claramente que el método de Newton-Raphson Mejorado arroja casi los mismos resultados, pero con mayor precisión, también se observa el parámetro "Tiempo" que dependerá de la velocidad del microprocesador que empleé la computadora utilizada, para este ejemplo es "cero" ya que desde que inicia el proceso hasta que termina el tiempo es el "mismo" y esto se debe a que el programa se ejecutó en una computadora con un microprocesador AMD™ Athlon™ de 1.33 GHz, lo cual fue tan rápido que no hubo diferencia alguna. El desarrollo de este programa fue hecho en Visual Basic 6.0®. El programa de cómputo se presenta en el CD de consulta, con el nombre de NRM.exe (consultar Apéndice A).

Cabe mencionar que para solución del sistema de ecuaciones lineales se ocupó el Método de Decomp y Solve, así como el método LU para la comparación de los resultados, lo cual se vio para este ejemplo que no había diferencia en su precisión.

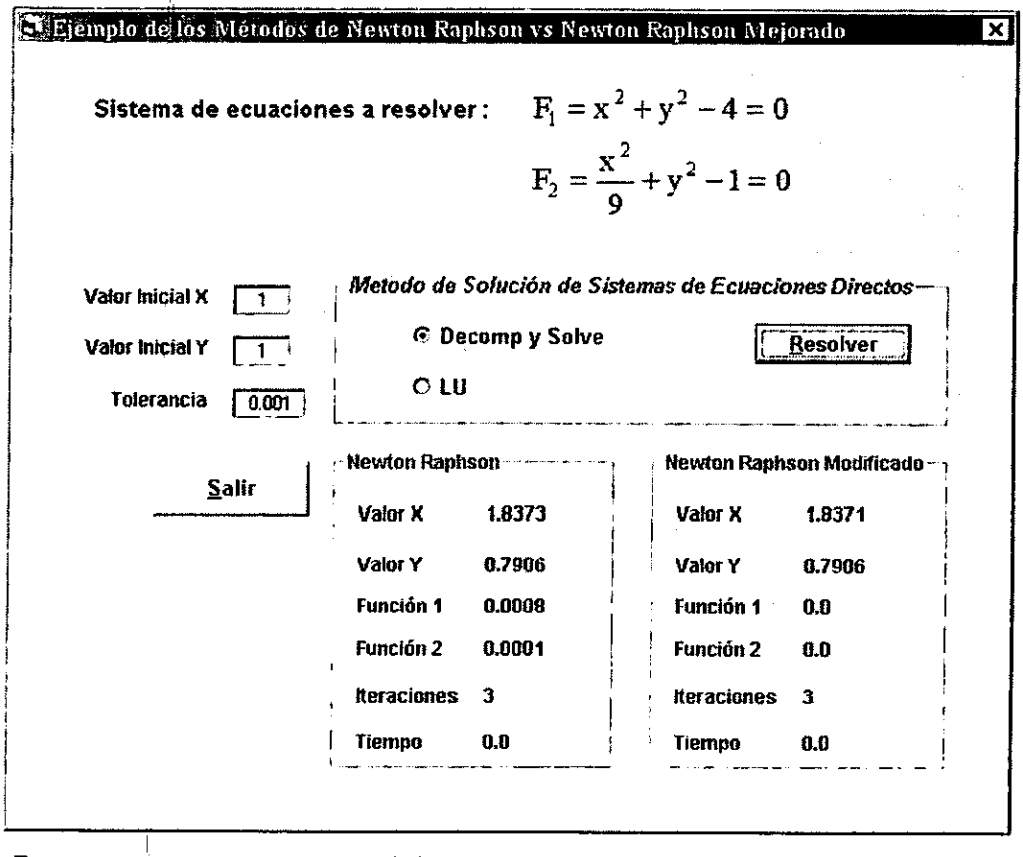


Figura 3.9. *RESULTADOS OBTENIDOS DE LOS MÉTODOS DE NEWTON-RAPHSON Y NEWTON-RAPHSON MEJORADO.*

En resumen, se puede mencionar que la ventaja de este método con respecto al de Newton-Raphson sin modificar, es que tiene una convergencia mucho más rápida y su radio de convergencia es mayor. Sin embargo, esta ventaja se obtiene a expensas de cálculos adicionales en cada iteración.

III.3.5 SOLUCIÓN AL PROBLEMA DE DISEÑO DE UN SISTEMA DE RECOLECCIÓN Y DISTRIBUCIÓN DE GAS.

Es factible establecer un modelo para el flujo de gas en un sistema de transporte, cuando las condiciones de flujo en cada uno de sus elementos, pueden describirse en términos de relaciones matemáticas. También es posible desarrollarlas ecuaciones analíticas necesarias para describir los efectos de interacción entre los componentes. El método más general y efectivo para analizar sistemas que manejan gas, es el propuesto por M. A. Stoner; en este subcapítulo se presentan los fundamentos del mismo y se discuten sus aplicaciones.

En la Figura 3.10, se muestra un sistema de recolección de gas natural constituido por una red de tuberías y otros elementos, tales como: compresores, pozos, reguladores, válvulas, etcétera, en régimen de flujo permanente

Cualquier red puede representarse como un sistema integral mediante nodos y conectores de nodos, como se ilustra en la Figura 3.11. Un nodo simboliza físicamente un elemento o punto de unión entre dos o más elementos, puede permitir adición o extracción de masa al sistema. Los conectores son elementos que permiten la transferencia de masa de un nodo a otro, por ejemplo, tuberías o válvulas.

Al conjunto de conectores se le denominará P, y N al de nodos. En un sistema como el descrito, la ley de conservación de masa en cada uno de los nodos debe satisfacerse, de manera analítica se tiene:

$$F_i = \sum_{j/(i,j)\in P} s_{ij}\, q_{ij} + Q_i = 0 \quad , \quad i \in N \, , \tag{3.99}$$

Esta ecuación establece que la masa que entra al nodo es igual a la que sale, por esta razón, algunas veces se le denomina Ecuación de balance en los nodos, y describe convenientemente la interacción de los diferentes elementos del sistema.

donde:

s_{ij} = variable que indica el sentido de flujo, positivo si el sentido de flujo es del nodo *"i"* al nodo *"j"*, en caso contrario se toma como negativo.

q_{ij} = gasto de gas que fluye a través del conector de los nodos *"i"* , *"j"*.

Q_i = término que indica adición de masa (Producción; negativo) o extracción de masa (Inyección; positivo) al sistema a través del nodo *"i"*.

Por ejemplo, considerando la Figura 3.11, la ecuación de conservación de masa para el nodo 1, es:

$$F_1 = s_{1,10}\, q_{1,10} + Q_1 = 0 \quad . \tag{3.100}$$

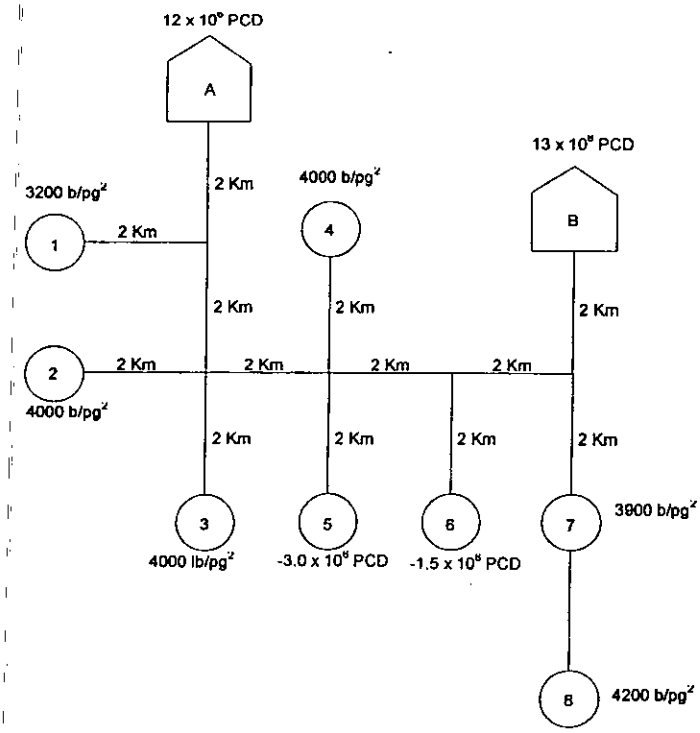


Figura 3.10. CAMPO "Las Margaritas", ESTUDIO DE UN DISEÑO DE RECOLECCIÓN DE GAS.

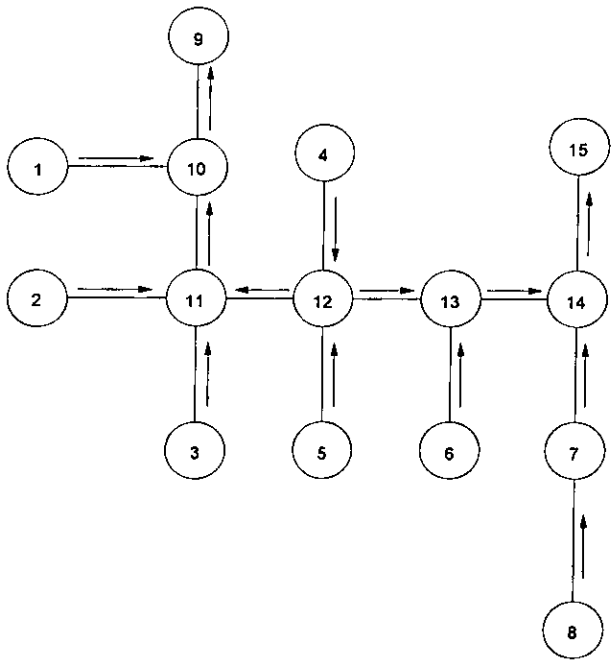


Figura 3.11. REPRESENTACIÓN POR MEDIO DE NODOS Y CONECTORES DEL SISTEMA DE RECOLECCIÓN DEL CAMPO "Las Margaritas". LAS FLECHAS INDICAN EL SENTIDO DE FLUJO A TRAVÉS DE LOS CONECTORES.

Cada gasto *"q_{ij}"* puede expresarse en términos de la diferencia de presión en los extremos del conector, o sea:

$$q_{ij} = C_{ij} \left| p_i^2 - p_j^2 \right|^n \quad , \tag{3.101}$$

donde:

C_{ij} = coeficiente de transmisión de la tubería, ver ecuación (3.104), que depende de la geometría de la tubo, de las condiciones de flujo y de la composición del gas.

p_i = presión del fluido en el nodo "i".

p_j = presión del fluido en el nodo "j".

n = exponente que depende de la forma de la ecuación de flujo a usar. Por ejemplo: para Panhandle A, n = 0.5392; Panhandle B, n = 0.51; Weymouth, n = 0.5.

Al sustituir la ecuación (3.101) en la (3.100), se obtiene:

$$F_i = \sum_{j/(i,j)\in P} s_{ij}\, c_{ij} \left| p_i^2 - p_j^2 \right| + Q_i = 0 \quad , \quad i \in N \, . \tag{3.102}$$

Como el sistema deber estar balanceado se cumple que:

$$\sum_{i=1}^{N} Q_i = 0 \quad . \tag{3.103}$$

De esta manera, el problema consiste en determinar un conjunto de valores Q_i y p_i, para toda *"i"* $\in$ N, que satisfaga las ecuaciones (3.102) y (3.103). Sin embargo, se presentan n ecuaciones con 2n incógnitas (n valores de p más n valores de Q), por lo tanto, se requiere asignar valores a n variables (valores medidos), siendo las n restantes, las incógnitas del sistema. Dicha asignación, se hará de forma que las ecuaciones resultantes sean linealmente independientes, entonces, como la suma algebraica de los gastos exteriores debe ser igual a cero, solamente se puede suponer gastos en (n-1) nodos, quedando al menos uno como incógnita.

Una posible forma para calcular la caída de presión en la tubería es mediante la ecuación de Panhandle[9]:

$$q_{ij} = 0.84269\, E \left[\frac{\left(p_i^2 / z_i\right) - \left(p_j^2 / z_j\right)}{0.6215\, L_{ij}} \right]^{0.5} d_{ij}^{2.667} \quad , \tag{3.104}$$

donde:

q_{ij} = gasto de gas a C.S., [MMPCD].

E = factor de eficiencia de la tubería, varía de 0.8 a 1.

p_i, p_j = presiones en los nodos i y j, respectivamente, [lb/pg^2].

z_i, z_j = factores de desviación del gas a p_i y p_j respectivamente.

L_{ij} = longitud de la línea entre los nodos i y j, [km].

d_{ij} = diámetro de la línea entre los nodos i y j, [pg].

De la expresión anterior y considerando la (3.101), se puede obtener el coeficiente de transmisión de la tubería (C_{ij}), esto es:

$$C_{ij} = \frac{0.84269\, E\, d_{ij}^{2.667}}{\left(0.6215\, L_{ij}\right)^{0.5}} \quad , \tag{3.105}$$

y para el sentido de flujo:

$$s_{ij} = \frac{p_i - p_j}{\left|p_i - p_j\right|} \quad . \tag{3.106}$$

Las ecuaciones que resultan de aplicar la ecuación (3.102) a cada uno de los nodos son no lineales. El método que se emplea para linealizar dichas ecuaciones es el de Newton-Raphson (sección III.3.3).

Como ocurre con todo proceso iterativo, la convergencia del método está sujeta a la selección de los valores iniciales para las incógnitas. Entre mejor sea la selección, el proceso convergerá en menos iteraciones. Para evitar la divergencia, Stoner[9] sugiere que en las dos primeras iteraciones, los valores de las correcciones se multipliquen por ½; aunque con esto, en las últimas etapas iterativas disminuye el ritmo de convergencia.

El simulador numérico desarrollado en este trabajo se estructuró en lenguaje *Visual Basic* 6.0 y permite manejar en forma gráfica el diseño de la red, mediante el desplazamiento del cursor en la pantalla del computador. El programa de cómputo se presenta en el CD de consulta, bajo el nombre de Redgas.exe (consultar Apéndice A).

Cabe mencionar que la matriz asociada al sistema de ecuaciones es *"rala"* y que su estructura no varía durante el proceso iterativo; no obstante, para fines didácticos se emplearán las subrutinas DECOMP y SOLVE, expuestas en la sección III.2.4, para resolver el sistema, aunque no sea la adecuada para este tipo de matrices, ya que se consume más tiempo del necesario.

En la página 89 se expone el diagrama de bloques simplificado (Figura 3.12).

DATOS DEL DISEÑO.

La prueba del simulador se hará empleando valores de una red de recolección de gas del campo Las Margaritas, Distrito Frontera Noreste, la cual está conectada a 8 pozos y 17 tuberías de distribución (ver Figura 3.10). Todos los diámetros interiores son de 2 pulgadas, excepto el de las líneas que une a los conectores (9,10) y (14,15), que es de 4. Tomando como base el arreglo de la Figura 3.11, en el punto "A" deben entregarse 12 MMPCD de gas y en el "B", 13 MMPCD. Se conoce la presión en la cabeza de 6 pozos y el gasto en los 2 restantes.

Es necesario determinar:

a) ¿A qué presión se entregará el gas en los puntos A y B?
b) ¿Cuál será el gasto Q_i y la presión p_i en cada nodo?
c) El número de iteraciones en que se llega a la solución.

Los valores iniciales a considerar son (por convención a los gastos de producción se les asigna signo negativo):

Q_1 = -2.9 [MMPCD]	Q_4 = -3.1 [MMPCD]	p_5 = 3990 [lb/pg^2]	p_{10} = 3100 [lb/pg^2]	p_{13} = 3750 [lb/pg^2]
Q_2 = -3.6 [MMPCD]	Q_7 = -1.2 [MMPCD]	p_6 = 3700 [lb/pg^2]	p_{11} = 3880 [lb/pg^2]	p_{14} = 3400 [lb/pg^2]
Q_3 = -3.6 [MMPCD]	Q_8 = -6.0 [MMPCD]	p_9 = 1000 [lb/pg^2]	p_{12} = 3900 [lb/pg^2]	p_{15} = 1050 [lb/pg^2]

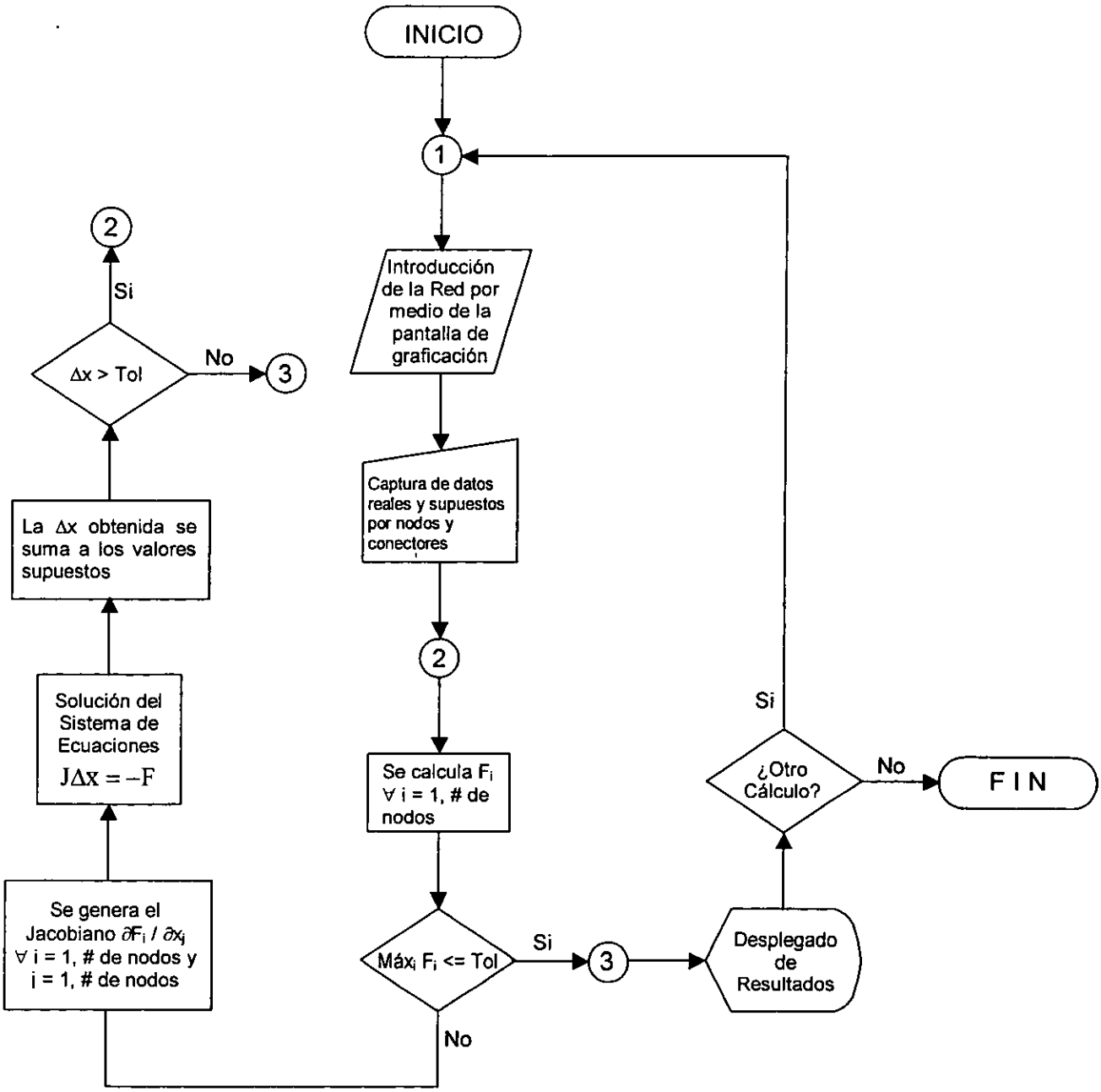


Figura 3.12. DIAGRAMA DE BLOQUES DEL SIMULADOR PARA FLUJO DE GAS EN RÉGIMEN PERMANENTE.

Todos los datos del sistema de recolección del presente análisis se incluyen en el programa de cómputo, así como su representación gráfica (red), a manera de ejemplo de aplicación.

RESULTADOS

Con fines de simplificación del problema, se supuso que el factor de desviación del gas (z) es igual a 1 en todo instante de la simulación, y que el factor de eficiencia es igual a 0.8.

En base a experimentación, se determinó una pequeña modificación al método de Stoner para acelerar la convergencia en redes de más de 5 nodos, ésta se basa en que en las tres primeras iteraciones se multiplique el vector de correcciones por 0.5. Con este criterio, se logró obtener la solución en 7 iteraciones en 69.64 segundos; considerando el método tradicional se ejecutan 12 iteraciones en 119.63 segundos, para alcanzar resultados análogos. En la Tabla 3.4 se plasman estos resultados.

nodo	p (lb/pg²)	Q (MPCD)
1	3200.00	-3200.10*
2	4000.00	-3605.55*
3	4000.00	-3605.55*
4	4000.00	-3131.66*
5	3993.15*	-3000.00
6	3769.29*	-1500.00
7	3900.00	-1168.66*
8	4200.00	-5986.49*
9	3063.79*	1,200.00
10	3103.05*	0.00
11	3888.26*	0.00
12	3916.00*	0.00
13	3749.00*	0.00
14	3426.17*	0.00
15	3384.46*	13,000.00

Tabla 3.4. RESULTADOS DEL SIMULADOR PARA EL CAMPO "Las Margaritas".

***valores resultantes para las incógnitas**

Debe indicarse que es importante elegir la ecuación correcta para evaluar las pérdidas de presión en tuberías, puesto que los resultados que proporciona el modelo están fincados totalmente en dicha selección. También, debe determinarse un factor de eficiencia que, unido a la ecuación, reproduzca las caídas de presión reales.

Puede observarse que simular sistemas de recolección y/o distribución de gas proporciona información esencial, como lo es: cuantificar el efecto que produciría cualquier cambio en el mismo, determinar su capacidad máxima, y estudiar el comportamiento del yacimiento, considerando la interacción entre los pozos.

III.4 PROBLEMAS RESUELTOS.

Problema Resuelto III.1. Sea el sistema:

$$\begin{bmatrix} 10 & -7 & 0 \\ -3 & 2.099 & 6 \\ 5 & -1 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 3.901 \\ 6 \end{bmatrix} .$$

Cuya solución exacta es $[0, -1, 1]^T$.

Empleando el método de Eliminación Gaussiana, investigue el efecto de los pivotes, sobre el error de precisión.

Suponga que se utiliza una computadora que maneja 5 cifras significativas.

Solución:

Conformando la matriz ampliada del sistema y eligiendo el valor del primer elemento del renglón 1 como pivote, se tiene:

1er. paso de la Eliminación Gaussiana:

- multiplicando la renglón 1 por 0.3 y sumándola a la renglón 2.
- multiplicando la renglón 1 por -0.5 y sumándola a la renglón 3.

$$\left[\begin{array}{ccc|c} 10 & -7 & 0 & 7 \\ 0 & -0.001 & 6 & 6.001 \\ 0 & 2.5 & 5 & 2.5 \end{array}\right].$$

Ahora, seleccionando al segundo elemento de la renglón 2 como pivote (recordemos que sólo se manejan 5 cifras significativas, por lo tanto, aquellas cantidades que rebasen esta restricción serán redondeadas):

2do. paso de la Eliminación Gaussiana:

- multiplicando la renglón 2 por $2.5x10^3$ y sumándola a la renglón 3, por lo que el nuevo renglón 3 es:

$2.5x10^3$ (-0.001) = -2.5 + 2.5 = 0 } matriz de coeficientes
$2.5x10^3$ (6) = $1.5005x10^4$ + 5 = $1.5005x10^4$ } matriz de coeficientes
$2.5x10^3$ (6.001) = $1.5003x10^4$ + 2.5 = $1.5006x10^4$ } vector de térm. indep.

$$\left[\begin{array}{ccc|c} 10 & -7 & 0 & 7 \\ 0 & -0.001 & 6 & 6.001 \\ 0 & 0 & 1.5005x10^4 & 1.5006x10^4 \end{array}\right].$$

Entonces, por sustitución hacia atrás:

$$x_3 = \frac{1.5006x10^4}{1.5005x10^4} = 1.0001 \quad ,$$

$$-0.001x_2 + 6x_3 = 6.001 \quad ,$$

$$x_2 = \frac{6.001 - 6(1.0001)}{-0.001} = -0.60013 \quad ,$$

$$10x_1 - 7x_2 = 7 \quad ,$$

$$x_1 = \frac{7 + 7(-0.4)}{10} = 0.2799 \quad .$$

La solución obtenida es $[0.2799, -0.60013, 1.0001]^T$, la cual difiere en extremo de la solución exacta $[0,-1,1]^T$. Si no existe error de redondeo acumulado, causado por la ejecución de miles de operaciones, y la matriz es no singular, ¿Cuál es el problema?. La dificultad se presenta al elegir un pivote de valor pequeño en el segundo paso de la eliminación. Esto introdujo el uso de un multiplicador de 2.5×10^3, por lo que, la ecuación final involucra coeficientes que son 10^3 veces mayores que los originalmente planteados. Se deja al lector la verificación de que al intercambiar el segundo renglón por el primero, no existirán multiplicadores grandes, y por ende, el resultado será exacto.

Se puede concluir que si se seleccionan pivotes con valores absolutos grandes, que impliquen multiplicadores con valor absoluto menor o igual 1, se obtendrán resultados satisfactorios. A este criterio se le denomina *Pivoteo Parcial.*

Problema Resuelto III.2. Elabore un programa de cómputo para obtener el determinante y la inversa de una matriz "cuadrada" cualquiera. Emplee las subrutinas DECOMP y SOLVE. Considere la siguiente matriz, para evaluar el funcionamiento del programa:

$$A=\begin{bmatrix}2 & -7 & 4\\ 1 & 9 & -6\\ -3 & 8 & 5\end{bmatrix} \quad ; \quad A^{-1}=\begin{bmatrix}\frac{93}{235} & \frac{67}{235} & \frac{6}{235}\\ \frac{13}{235} & \frac{22}{235} & \frac{16}{235}\\ \frac{7}{47} & \frac{1}{47} & \frac{5}{47}\end{bmatrix} .$$

Solución:

Mediante la subrutina DECOMP se puede obtener el determinante de una matriz (A), para ello, se recurre a la siguiente relación, la cual involucra elementos calculados por la subrutina mencionada:

$$\det(A) = IPVT(N) \times a(1,1) \times a(2,2) \times ..... \times a(N, N)$$

Por otro lado, la inversa de la matriz de coeficientes (A), puede definirse como la matriz X, cuyas columnas $\bar{x}_j$, satisfacen:

$$A\,\bar{x}_j = \bar{e}_j \quad , \quad j=1,2,.....,n \quad ,$$

donde:

$\bar{e}_j$ = es la j-ésima columna de la matriz identidad (I).

Entonces, la matriz inversa (X), se obtendrá llamando a DECOMP en una sola ocasión y a SOLVE "n" veces, una para cada columna de "X". Si se detecta singularidad, "X" será indeterminable[1]. El programa de cómputo se presenta en el CD de consulta, y se intitula Inversa.exe. (consultar Apéndice A). Los resultados arrojados por él, son los siguientes:

$$X=\begin{bmatrix}0.3957 & 0.2851 & 0.0255\\ 0.0553 & 0.0936 & 0.0681\\ 0.1489 & 0.0213 & 0.1064\end{bmatrix} \quad , \quad \det(A)=235 \quad .$$

Como puede observarse, existe coincidencia total entre los elementos de la matriz inversa calculada (x) y la dada como dato (A^{-1}), debe aclararse también, que se redondeo a cuatro decimales significativos el resultado mostrado.

Problema Resuelto III.3. Dado el siguiente código en Visual Basic, explicar ¿por que el valor de A(2,2) después de llamar a DECOMP está relacionado con el machine epsilon de la computadora?. En el de consulta se presentan los programas de cómputo EstimaMatrizA.exe en Visual Basic y el Estima.for en Fortran.

```
Sub Estimar_Matriz_A()
Dim NDIM
NDIM = 10
n = 2
a(1, 1) = 3# : a(1, 2) = 6# : a(2, 1) = 1# : a(2, 2) = 2#
Call decomp(NDIM, n, a, COND, IPVT, WORK)
Form1.Label1.Caption = a(2,2)
Form1.Label2.Caption = COND
```

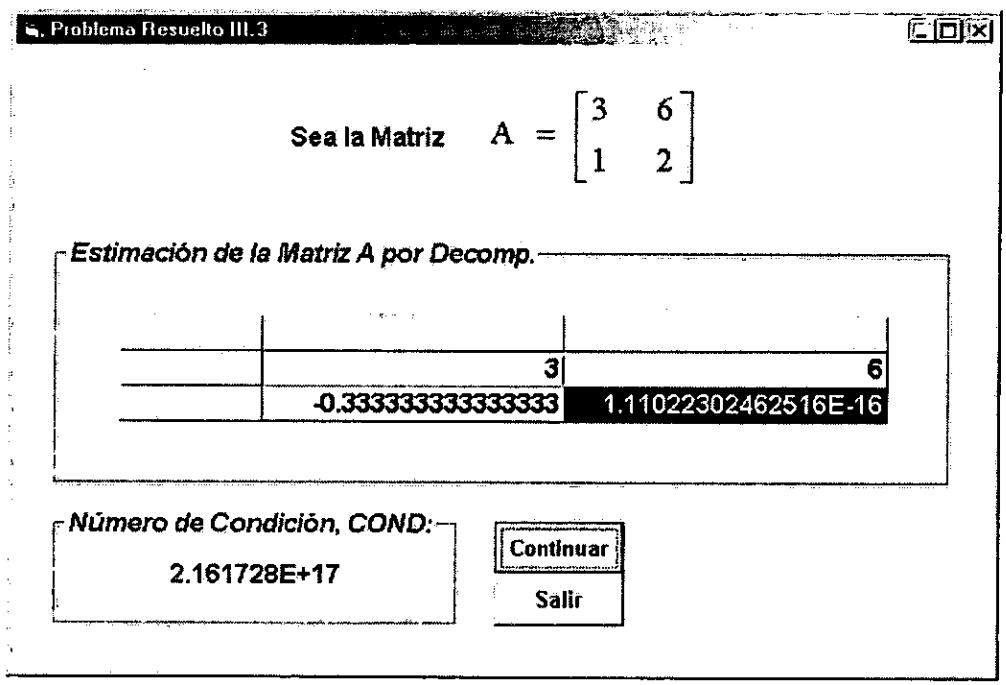


Figura 3.13. VENTANA DE RESULTADOS DE LA PROGRAMACIÓN EN VISUAL BASIC.

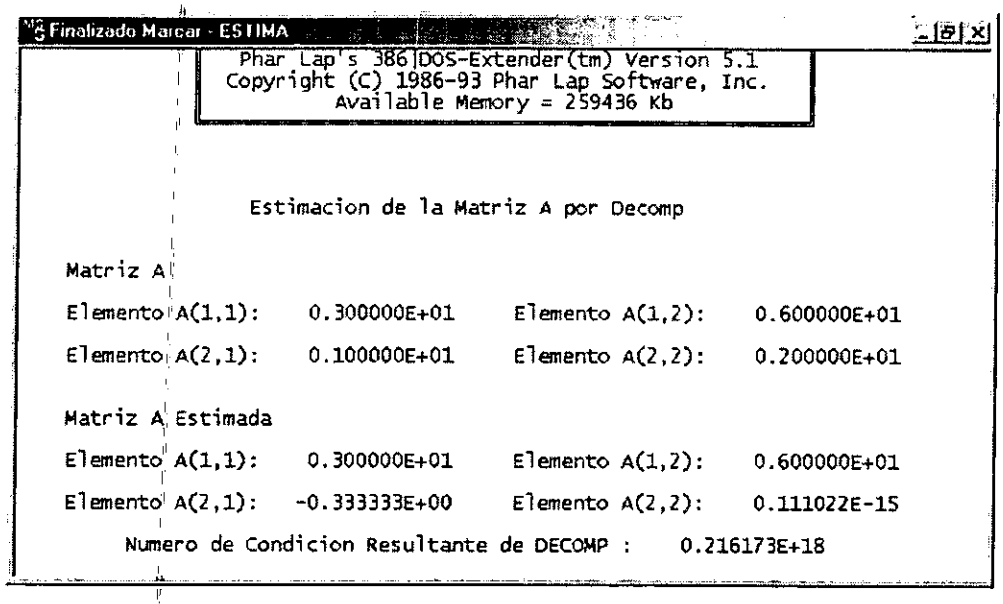


Figura 3.14. VENTANA DE RESULTADOS DE LA PROGRAMACIÓN EN FORTRAN.

Debido a que el número 1/3 no puede representarse en forma exacta en las computadoras digitales que manejan una $\beta = 2$, se presenta un error de redondeo al multiplicar este valor por 6 y el resultado sumarlo a 2.

El machine epsilon (ε) que esta definido en la sección II.4.1 es muy parecido en orden de magnitud cuando se utilizan definiciones que consideran en lugar del valor unitario los números tales como 2, 3, 4,..., etc. como se presentan en la tabla siguiente.

Número	Machine Epsilon (ε)
1	$1.11022302462516 \times 10^{-16}$
2	$2.22044604925031 \times 10^{-16}$
3	$1.66533453693773 \times 10^{-16}$
4	$4.44089209850063 \times 10^{-16}$
5	$2.77555756156289 \times 10^{-16}$
6	$3.33066907387547 \times 10^{-16}$
7	$3.88578058618805 \times 10^{-16}$
8	$8.88178419700125 \times 10^{-16}$
9	$4.9960036108132 \times 10^{-16}$
10	$5.55111512312578 \times 10^{-16}$

Tabla 3.5. MACHINE EPSILON UTILIZANDO VARIOS NÚMEROS.

Lo mencionado anteriormente responde al cuestionamiento del porqué el valor de A(2,2) está relacionado con el valor del machine epsilon (ε)

III.5 PROBLEMAS PROPUESTOS.

Problema Propuesto III.1. Se requiere diseñar la sección de la torre de perforación que se muestra en la Figura 3.15. Por lo tanto, deben calcularse las fuerzas que actúan en cada barra, con el propósito de determinar el diámetro de éstas. En la parte superior, una fuerza de 20 toneladas simula el efecto de tensión, debido a la extracción o introducción de tubería. En la parte media, una fuerza de 5 toneladas representa el efecto originado por el apoyo de la tubería contra la estructura. En la parte inferior, una fuerza de 10 toneladas simula el peso de la estructura metálica.

Si las fuerzas horizontales se denotan como Fx y las verticales como Fy, además, considerando condiciones estáticas (equilibrio), el problema puede representarse de la manera siguiente:

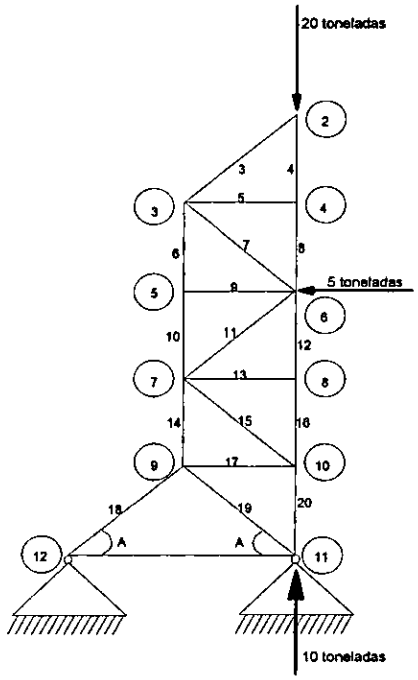


Figura 3.15. DISEÑO DE TORRE DE PERFORACIÓN.

$$\text{junta 2}\begin{cases}\sum Fy = f_4 + rf_3 - 20 = 0\end{cases}$$

$$\text{junta 3}\begin{cases}\sum Fx = rf_3 + f_5 + rf_7 = 0\\ \sum Fy = f_6 + rf_7 - rf_3 = 0\end{cases}$$

$$\text{junta 4}\begin{cases}\sum Fx = f_5 = 0\\ \sum Fy = f_8 - f_4 = 0\end{cases}$$

$$\text{junta 5}\begin{cases}\sum Fx = f_9 = 0\\ \sum Fy = f_{10} - f_6 = 0\end{cases}$$

$$\text{junta 6}\begin{cases}\sum Fx = f_9 + rf_7 + rf_{11} = 0\\ \sum Fy = f_{12} + rf_{11} - rf_7 - f_8 = 0\end{cases}$$

$$\text{junta 7}\begin{cases}\sum Fx = f_{13} + rf_{11} + rf_{15} = 0\\ \sum Fy = f_{14} + rf_{15} - rf_{11} - f_{10} = 0\end{cases}$$

$$\text{junta 8}\begin{cases}\sum Fx = f_{13} = 0\\ \sum Fy = f_{16} - f_{12} = 0\end{cases}$$

$$\text{junta 9}\begin{cases}\sum Fx = rf_{18} - rf_{19} - f_{17} = 0\\ \sum Fy = rf_{18} + rf_{19} - f_{14} = 0\end{cases}$$

$$\text{junta 10}\begin{cases}\sum Fx = f_{17} + rf_{15} = 0\\ \sum Fy = f_{20} - rf_{15} - f_{16} = 0\end{cases}$$

$$\text{junta 11}\begin{cases}\sum Fy = 10 - rf_{19} - f_{20} = 0\end{cases}$$

Realice un programa que involucre a las subrutinas DECOMP y SOLVE, para resolver el problema planteado.

Problema Propuesto III.2. La importancia de poseer una herramienta que permita simular el comportamiento de un sistema de conducción de fluidos en dos fases, con régimen estacionario, a través de tuberías, es incalculable, ya que con ella se pueden diseñar redes para la recolección y distribución de fluidos, así como optimizarlos sistemas existentes. En base al simulador de flujo de gas en régimen permanente, elabore un programa de cómputo que simule el flujo bifásico en tuberías horizontales, empleando el método propuesto por Begss y Brill (referencia: Brill, J. P. y Beggs,H. D., Two phase flow in Pipes, abril 1979).

INTERPOLACIÓN Y APROXIMACIÓN NUMÉRICAS

IV

IV.1 INTRODUCCIÓN.

En cualquier rama de la Ciencia, sea ésta: Ingeniería, Matemáticas Aplicadas, etcétera, es necesario lograr soluciones, prácticas y rápidas de fenómenos o procesos físicos y químicos. En ocasiones, tales eventos sólo se conocen de manera tabulada o gráfica, es entonces, cuando la posibilidad de interpolar los datos o de generar una función de aproximación se torna primordial, pues con ello se obtiene una solución referida a los parámetros en estudio. Esta técnica evita el reproducir el proceso o fenómeno para cualquier condición requerida, con el consecuente ahorro de tiempo e inversión.

IV.2 INTERPOLACIÓN EXACTA.

Suponga que se cuenta con un conjunto de *"n"* pares de puntos (puntos base), definidos en el conjunto $\Re^2$, tal que:

$$(x_1, y_1), (x_2, y_2), ..., (x_n, y_n) \quad , \quad x_1 < x_2 < ..... < x_n$$

Los valores de las ordenadas (y_i), son el resultado de alguna observación experimental o de alguna relación matemática y corresponden a las condiciones x_i. Si se desea conocer (interpolar) el valor de "y" para $x \neq x_i$, deberá crearse un función f, de tal manera que:

$$f(x_i) = y_i \quad , \quad i = 1, 2, ..., n . \tag{4.1}$$

Además, la citada función debe proporcionar valores razonables para f(x), $x \neq x_i$. El criterio para definir valor razonable varía de acuerdo al problema analizado y está supeditado a los requerimientos del analista. Si las ordenadas (y_i), conforman una función matemática *suave* y poseen un cierto número de cifras decimales exactas, se espera que la solución sea satisfactoria.

Si los puntos (x_i, y_i), se derivan de observaciones experimentales muy precisas, de forma que pueden considerarse *libres* de error, es apropiado interpolarlos mediante una función suave, como la que ofrece la técnica de los Splines Cúbicos (que se expondrá posteriormente). Si por el contrario, provienen de experimentos relativamente burdos o imprecisos, es absurdo *obligar* a que la función interpolante se ajuste exactamente a todos los puntos dato. En tal caso, será suficiente con ajustar una función global para lograr resultados aceptables. Como se describirá más adelante.

El punto clave de la interpolación, es definir como debe comportarse una *función razonable* a través de los puntos dato. Debido a que los puntos base pueden interpolarse con un número infinito de funciones distintas, es necesario emitir algún criterio para elegir alguna de ellas. El criterio típico se sustenta en la suavidad y simplicidad de la función a seleccionar.

La suavidad de una función está dada por el valor máximo de $|f''(x)|$. Entre menor sea dicho valor, la suavidad irá en aumento. La simplicidad se refiere al grado de la función y también debe ser el mínimo valor posible.

Aunque existen diversas funciones de interpolación -monomiales, polinomiales, trigonométricas o de Fourier, exponenciales y racionales- se detallarán sólo las polinomiales, por ser las más empleadas (consultar sección IV.3, Aproximación Funcional).

IV.2.1 INTERPOLACIÓN POLINOMIAL.

Dadas las parejas de valores (x_i, $f(x_i) = y_i$), i = 0, 1, 2, ..., n y el polinomio de interpolación

$$p_n(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n . \qquad (4.2)$$

El criterio más evidente para determinar los coeficientes de $p_n(x)$, es el satisfacer la siguiente expresión:

$$p_n(x_i) = f(x_i) \quad , \quad i = 1, 2, ..., n . \qquad (4.3)$$

Entonces, el polinomio $p_n(x)$ de n-ésimo grado debe reproducir exactamente a f(x), para los n+1 argumentos $x = x_i$. Este razonamiento, basado en el teorema de Weierstrass, es especialmente importante, ya que existe un polinomio único de grado "n" o menor, que cumpla tal aseveración[1]. Un polinomio con esas características es denominado *polinomio de interpolación de n-ésimo grado.*

Nótese, que la ecuación (4.3) establece que debe satisfacerse el valor de $p_n(x)$, para toda x_i, pero de ninguna manera garantiza la aproximación exacta de f(x) para $x \neq x_i$, es decir, para argumentos diferentes de los puntos base. En la Figura 4.1 se presenta de manera esquemática un polinomio interpolador para n = 3.

Aunque la mayoría de las fórmulas de interpolación que la literatura brinda, aparentemente sean disímbolas, aquellas que empleen como información idénticos puntos base y el criterio de la ecuación (4.3), para calcular los coeficientes $a_0, a_1, ..., a_n$, deben ser básicamente iguales.

La interpolación polinomial puede representarse por un sistema de ecuaciones lineales, o sea:

$$\begin{array}{ccccccccc}
a_0 & + & a_1 x_0 & + & a_2 x_0^2 & + & \cdots & + & a_n x_0^n = f(x_0) \\
a_0 & + & a_1 x_1 & + & a_2 x_1^2 & + & \cdots & + & a_n x_1^n = f(x_1) \\
a_0 & + & a_1 x_2 & + & a_2 x_2^2 & + & \cdots & + & a_n x_2^n = f(x_2) . \\
\vdots & & \vdots & & \vdots & & \vdots & & \vdots \\
a_0 & + & a_1 x_n & + & a_2 x_n^2 & + & \cdots & + & a_n x_n^n = f(x_n)
\end{array} \qquad (4.4)$$

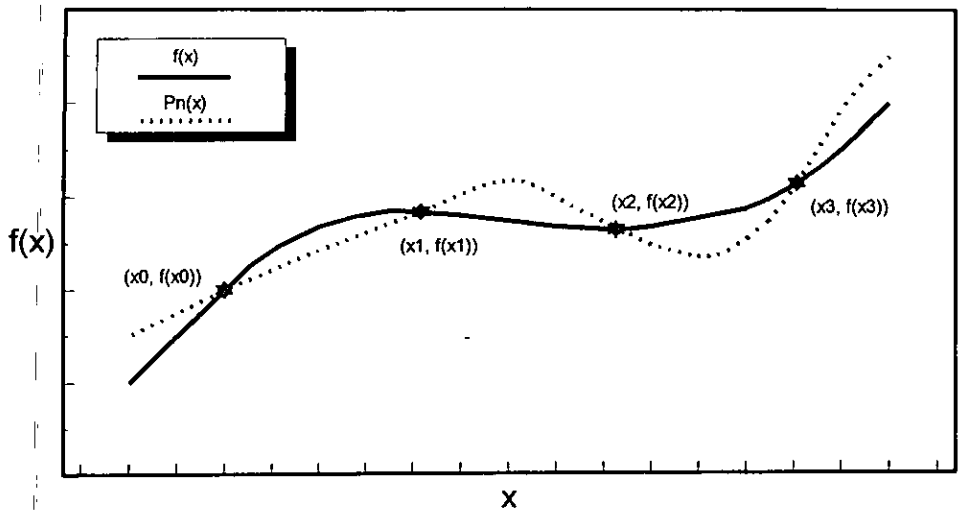


Figura 4.1. INTERPOLACIÓN POLINOMIAL. Para 4 pares de puntos dato.

El determinante de la matriz de coeficientes, el cual se conoce como determinante de Vandermonde, es:

$$\begin{vmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ & & \vdots & & \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{vmatrix}. \tag{4.5}$$

Se observa que para $x \neq x_i$, $i \neq j$, este determinante es diferente de cero, por lo tanto, existe una solución única para cada a_i, es decir, habrá un polinomio único $p_n(x)$, que se ajuste con exactitud a f(x) en los puntos dato.

Ya que los coeficientes del polinomio elegido pueden obtenerse al resolver el sistema de ecuaciones inherente a él, surge la pregunta: ¿Para qué emplear fórmulas de interpolación?. Existen dos razones fundamentales. La primera, es que dar solución a sistemas de ecuaciones lineales de cualquier tamaño, no es una tarea sencilla, sobre todo, cuando se *hace a mano*. La segunda, y quizá la más importante, es que la matriz de coeficientes es mal condicionada. Un ejemplo clásico, es el de los valores x_i igualmente espaciados en el intervalo [0, 1], que al ser sustituidos en las funciones 1, x, x^2,...,x^n, producen elementos

positivos entre 0 y 1, generándose una estrecha dependencia entre las columnas o renglones de la matriz de coeficientes.

IV.2.2 INTERPOLACIÓN POLINOMIAL DE LAGRANGE.

La ecuación de interpolación de Lagrange es la siguiente:

$$\begin{aligned} p_n(x) = {} & a_0(x-x_1)(x-x_2)(x-x_3)\cdots(x-x_n)+ \\ & +a_1(x-x_0)(x-x_2)(x-x_3)\cdots(x-x_n)+ \\ & +a_2(x-x_0)(x-x_1)(x-x_2)\cdots(x-x_n)+ \\ & \vdots \\ & +a_i(x-x_0)(x-x_1)\ldots(x-x_{i-1})\,(x-x_{i+1})\cdots(x-x_n)+ \\ & +a_{n-1}(x-x_0)(x-x_1)\cdots(x-x_{n-2})(x-x_n)+ \\ & +a_n(x-x_0)(x-x_1)\cdots(x-x_{n-2})(x-x_n) \end{aligned} \tag{4.6}$$

En la ecuación (4.6), los coeficientes a_0, a_1,...,a_n, serán determinados de tal forma que: $p_n(x_i) = f(x_i)$, i = 0, 1, 2,...,n.

Por definición los coeficientes son:

$$a_i = \frac{f(x_i)}{(x_i-x_0)(x_i-x_1)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)} . \tag{4.7}$$

La forma condensada de la expresión de Lagrange se obtiene al sustituir la ecuación (4.7) en la (4.6), o sea:

$$p_n(x) = \sum_{i=0}^{n} L_i(x) f(x_i) , \tag{4.8}$$

donde:

$$L_i(x) = \prod_{\substack{j=c \\ j\neq i}}^{n} \frac{(x-x_j)}{(x_i-x_j)} \quad , \quad i = 0,1,....,n . \tag{4.9}$$

En la expresión (4.8), cada valor de $f(x_i)$ es multiplicado por $L_i(x)$, el cual es un polinomio de ajuste de n-ésimo grado, ya que está constituido por "n" factores $(x - x_j)$.

Una desventaja de esta técnica, es la gran cantidad de operaciones que deben realizarse cuando se desean varias interpolaciones con el mismo conjunto de puntos dato (el número de operaciones y el tiempo de ejecución son proporcionales a n^2). Otra desventaja, se presenta cuando se pretende incrementar en uno, el grado del polinomio (adición de un nuevo término), puesto que se requiere el cálculo completo de todos los valores $L_i(x)$. Por lo tanto, el polinomio de interpolación de Lagrange no es recomendable cuando el grado del polinomio no se conoce *a priori*.

IV.2.3 VALUACIÓN DE POLINOMIOS.

En algunos casos, en un programa de cómputo se requiere valuar reiteradamente un polinomio para cierto número de argumentos. Es entonces, que se hace vital la rápida valuación de dicho polinomio. Considérese el siguiente polinomio:

$$p_n(x) = a_1 x^n + a_2 x^{n-1} + ... + a_{n+1} \ . \qquad (4.10)$$

Cuya valuación en lenguaje *Basic* es:

```
P = A(N + 1)
FOR I = 1 TO N
      P = P + A(I) * x^(N - I + 1)
NEXT I
```

Usualmente se emplean n(n+1)/2 multiplicaciones y "n" adiciones.

Una técnica simple para optimizar la valuación de un polinomio es la denominada *regla de Horner*, conocida también como *división* o *sustitución sintética*, que consiste en un re-arreglo del polinomio original, esto es:

$$p_n(x) = a_{n+1} + x(a_n + x(a_{n-1} + x(...(a_2 + a_1 x)...))) \ . \qquad (4.11)$$

Que en lenguaje *Visual Basic* se puede valuar como:

```
P = A(1)
FOR I = 1 TO N
    P = P * x + A(I + 1)
NEXT I
```

El algoritmo anterior realiza solamente "n" multiplicaciones y adiciones. Esta técnica lleva el nombre de W. G. Horner, porque él la publicó en 1819, sin embargo, fue propuesta 100 años antes por Isaac Newton.

IV.2.4 SUBRUTINA SPLINE[1].

Antes de describir las *bondades* de la técnica *SPLINE*, se mostrará su ámbito teórico. Las funciones cúbicas *Spline* constituyen una técnica interpoladora reciente y se caracterizan por ser continuas, además, su primera y segunda derivadas también lo son.

Las funciones cúbicas Spline ajustan n-1 polinomios de tercer grado, es decir, un polinomio para cada uno de los n-1 intervalos definidos. Esta es la principal diferencia con respecto a las técnicas de interpolación tradicionales, en las cuales se ajusta un polinomio único a los "n" pares de puntos base (Figuras 4.2 y 4.3).

La teoría de los *Splines* se expone a continuación:

Sean las abscisas $(a = x_0) < x_1 < ... < (x_n = b)$ y las ordenadas $[y_i]$, i = 0, 1, 2,...,n. Puede demostrarse[3] que de todas las funciones f(x) con segunda derivada continua en el intervalo [a, b], tales que $f(x_i) = y_i$, la función cúbica *Spline* s(x), con segunda derivada igual a cero en los extremos del citado intervalo, $s''(a) = s''(b) = 0$, minimiza la integral:

$$\int_a^b (f''(x))^2 dx \ , \tag{4.12}$$

esto es:

$$\int_a^b (s''(x))^2 dx \le \int_a^b (f''(x))^2 dx \ . \tag{4.13}$$

La función cúbica *Spline* con condición s´´(a) = s´´(b) = 0, es denominada *Spline natural* y posee la menor curvatura (mayor suavidad) de entre todas las funciones que pueden interpolar los puntos dato.

En los n-1 intervalos, se tiene igual número de secciones separadas de curvas cúbicas, cada una con 4 parámetros, por lo que habrá 4n-4 elementos a determinar. El hecho de que s(x) sea continua, y que su primera y segunda derivadas también lo sean en los n-2 nodos interiores x_i, adiciona 3(n-2) condiciones. Como $s(x_i) = y_i$ en cada uno de los nodos, se introducen "n" condiciones más, siendo 4n-6 el número total de parámetros a determinar. Para generar un sistema compatible, deben incluirse dos condiciones extra, que pueden estar dadas por las condiciones de frontera, s´´(a) = s´´(b) = 0.

Construir una función *Spline* es un proceso simple y numéricamente estable. Supóngase el subintervalo (x_i, x_{i+1}), y que:

$$h_i = x_{i+1} - x_i \ ,$$
$$w = \frac{x - x_i}{h_i} \ ,$$
$$\overline{w} = 1 - w \ .$$

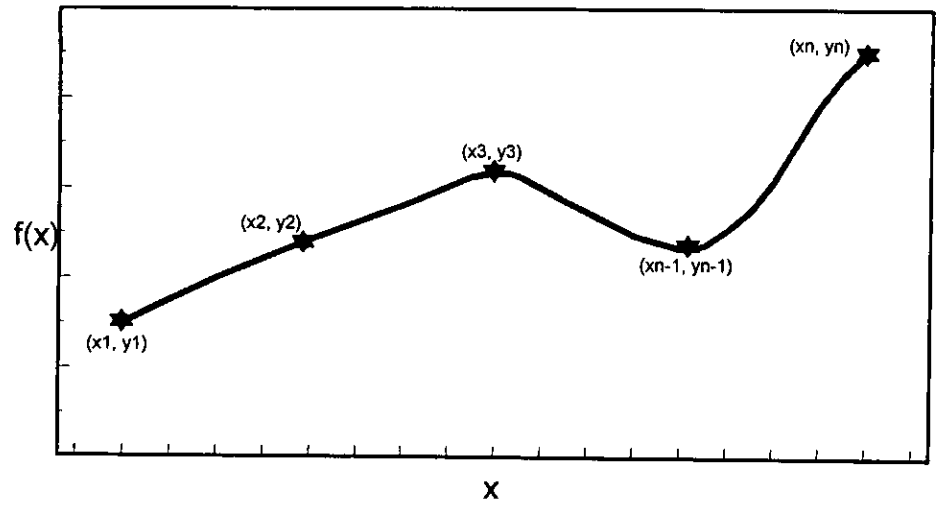


Figura 4.2. FUNCIÓN POLINÓMICA ÚNICA DE GRADO "n-1".

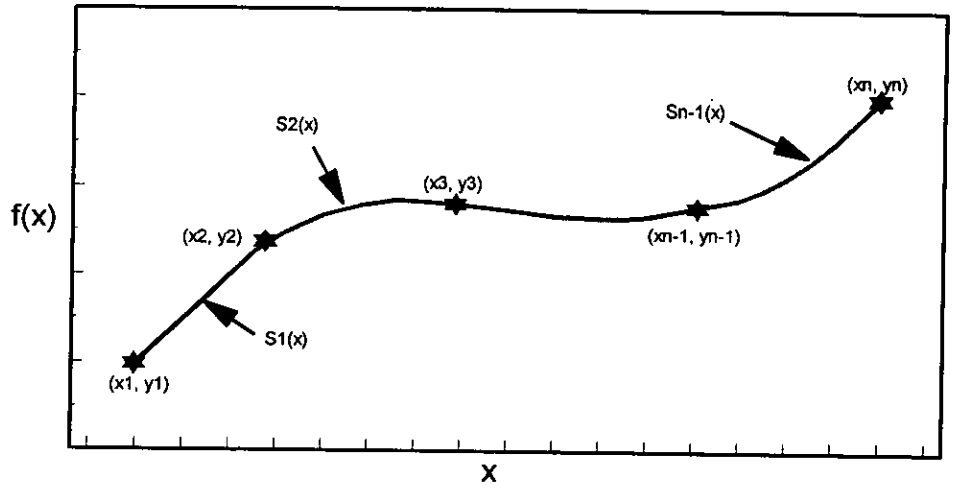


Figura 4.3. n-1 FUNCIONES CÚBICAS SPLINE AJUSTADAS EN n-1 INTERVALOS.

Debido a que x fluctúa en tal subintervalo, w variará de 0 a 1 y $\overline{w}$ de 1 a 0. Ahora, representando la función Spline por medio de:

$$s_i(x) = wy_{i+1} + \overline{w}y_i + h_i^2\left[\left(w^3 - w\right)\sigma_{i+1} + \left(\overline{w}^3 - \overline{w}\right)\sigma_i\right], \qquad (4.14)$$

donde:

σ_i y σ_{i+1} son constantes por determinar.

Los dos primeros términos de la expresión anterior representan una interpolación lineal, mientras que los términos entre paréntesis rectangulares realizan una corrección cúbica, que generará la suavidad en la solución. Obsérvese que el término corrector desaparece en los puntos inicial y final del subintervalo, de manera que:

$$s(x_i) = y_i \quad , \tag{4.15}$$

$$s(x_{i+1}) = y_{i+1} \quad . \tag{4.16}$$

Según esto, la función s(x) interpola exactamente los datos, sin importar cuales sean los valores de σ_i.

Ahora, diferenciando en tres ocasiones la función s(x) y utilizando la regla de la cadena, y considerando que:

$$w' = \frac{1}{h_i} \quad , \quad \overline{w}' = \frac{-1}{h_i} \ ,$$

se obtiene:

$$s'(x) = \frac{(y_{i+1} - y_i)}{h_i} + h_i\left[(3w^2 - 1)\sigma_{i+1} - (3\overline{w}^2 - 1)\sigma_i\right], \tag{4.17}$$

$$s''(x) = 6w\sigma_{i+1} + 6\overline{w}\sigma_i \ , \tag{4.18}$$

$$s'''(x) = \frac{6(\sigma_{i+1} - \sigma_i)}{h_i} \quad . \tag{4.19}$$

Debe señalarse que s´´(x) es una función lineal, la cual interpola entre los valores $6\sigma_{i+1}$, y $6\sigma_i$. Consecuentemente:

$$\sigma_i = \frac{s''(x_i)}{6} \ . \tag{4.20}$$

Lo que explica el significado de σ_i, pero, no determina su valor. Además, s´´´(x) es constante en cada subintervalo y s´´´´(x) = 0, puesto que s(x) es una función cúbica. Si se evalúa s´(x) en los puntos extremo del subintervalo se tiene:

$$s'_+(x_i) = \Delta_i - h_i(\sigma_{i+1} + 2\sigma_i) , \tag{4.21}$$

y:

$$s'_-(x_{i+1}) = \Delta_i + h_i(2\sigma_{i+1} + \sigma_i) , \tag{4.22}$$

donde:

$$\Delta_i = \frac{y_{i+1} - y_i}{h_i} . \tag{4.23}$$

En las expresiones anteriores se emplea de manera temporal s'_+ y s'_-, ya que la fórmula para s(x) se cumple sólo en el intervalo $[x_i, x_{i+1}]$ y las derivadas en los puntos extremo no están claramente definidas. Con la finalidad de lograr la continuidad deseada en s´(x), se fijan las siguientes condiciones en los puntos interiores:

$$s'_-(x_i) = s'_+(x_i) \quad , \quad i = 2, 3, ..., n-1 . \tag{4.24}$$

Aún cuando s'_- se calcule al considerar el subintervalo $[x_{i-1}, x_i]$, su fórmula puede obtenerse al reemplazar "i" por i-1 en $s'_-(x_{i+1})$, lo que origina:

$$\Delta_{i-1} + h_{i-1}(2\sigma_i + \sigma_{i-1}) = \Delta_i - h_i(\sigma_{i+1} + 2\sigma_i) , \tag{4.25}$$

reordenando:

$$h_{i-1}\sigma_{i-1} + 2(h_{i-1} + h_i)\sigma_i + h_i\sigma_{i+1} = \Delta_i - \Delta_{i-1} \quad ; \quad i = 2, 3, ..., n-1 . \tag{4.26}$$

La expresión anterior genera un sistema de n-2 ecuaciones lineales con "n" incógnitas, σ_i, i = 1, 2,...,n. Entonces, deben adicionarse dos condiciones para lograr una solución única. Tal problema se resuelve considerando un Spline natural, puesto que $s''(x_i) = s''(x_n) = 0$, implicando que $\sigma_i = \sigma_n = 0$.

Un Spline con estas condiciones frontera define un sistema de n-2 ecuaciones lineales y n-2 incógnitas, de forma que:

$$\begin{bmatrix} 2(h_1+h_2) & h_2 & 0 & 0 & 0 \\ h_2 & 2(h_2+h_3) & h_3 & 0 & 0 \\ 0 & \vdots & \vdots & \vdots & 0 \\ 0 & 0 & h_{n-3} & 2(h_{n-3}+h_{n-2}) & h_{n-2} \\ 0 & 0 & 0 & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{bmatrix} \begin{bmatrix} \sigma_2 \\ \sigma_3 \\ \vdots \\ \sigma_{n-2} \\ \sigma_{n-3} \end{bmatrix} = \begin{bmatrix} \Delta_1 - \Delta_2 \\ \Delta_3 - \Delta_2 \\ \vdots \\ \Delta_{n-2} - \Delta_{n-3} \\ \Delta_{n-1} - \Delta_{n-2} \end{bmatrix} . \quad (4.27)$$

Este sistema puede resolverse mediante las subrutinas DECOMP y SOLVE, no obstante, la matriz de coeficientes presenta las siguientes características especiales:

1) Es tri-diagonal.
2) Es simétrica.
3) Para cualquier $x_1 < x_2 < ... < x_n$, la matriz es no singular y generalmente esta bien condicionada.

Por lo tanto, emplear el algoritmo de Thomas reduce el tiempo de ejecución y la cantidad de operaciones por realizar.

Si el Spline se va a evaluar muchas veces, es recomendable calcular y almacenar los coeficientes del Spline cúbico, b_i, c_i y d_i, i = 1, 2,...,n-1, para cada intervalo $[x_i, x_{i+1}]$, siendo:

$$s(x) = y_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \quad ; \quad x_i \le x \le x_{i+1} . \quad (4.28)$$

Dichos coeficientes, pueden obtenerse con las ecuaciones siguientes:

$$b_i = \frac{y_{i+1} - y_i}{h_i} - h_i(\sigma_{i+1} + 2\sigma_i) , \tag{4.29}$$

$$c_i = 3\sigma_i , \qquad i = 1, 2,....,n-1 . \tag{4.30}$$

$$d_i = \frac{\sigma_{i+1} - \sigma_i}{h_i} . \tag{4.31}$$

De esta manera se simplifican manipulaciones de s(x), tales como derivaciones e integraciones.

La subrutina SPLINE tiene implementada la técnica descrita, para calcular los coeficientes de la función cúbica del mismo nombre. Los comentarios que aparecen en la rutina (consultar el anexo, archivo Spline.exe), describen la forma de usarla. Una vez calculados los coeficientes por SPLINE, el subprograma SEVAL valúa el Spline correspondiente al dato a interpolar. Tal subprograma, con formato de función (*function*), cuenta con una innovación (desarrollada en este trabajo), que permite elegir el tipo de búsqueda a ejecutar, sea ésta binaria o secuencial, cuando la variable independiente a interpolar no se encuentra en el mismo intervalo de la llamada previa, con la finalidad de ubicar el intervalo apropiado.

Por último, es pertinente apuntar que los valores de la variable independiente (abscisas), deben proporcionarse en estricto orden ascendente ($x_1 < x_2 <...< x_n$), para asegurar que los valores arrojados por SPLINE sean confiables y correctos.

IV.2.4.1 INTERPOLACIÓN SPLINE EN TRES DIMENSIONES.

Si se tienen "n" datos en el eje de las abscisas y "m" en el de las ordenadas, cada uno de ellos con su respectivo valor en el eje axial (Z), de tal forma que las coordenadas de cada punto sean:

$$(x_i, y_j, z_{ij}) \quad , \quad i = 1, 2,....,n \quad ; \quad j = 1, 2,....,n .$$

Los (m × n) puntos base definen una región rectangular de interpolación en el plano X-Y. El problema será entonces definir el valor de z^* en cualquier punto (x^*, y^*). El procedimiento es el siguiente:

a) Para cada y_j seleccionar las parejas de puntos (x_i, z_{ij}) correspondientes y obtener las (n-1 x m) funciones Spline $s_{kj}(x)$, k = 1, 2,...,n-1, involucradas.

b) Se elige el conjunto de funciones $s_{kj}(x)$, j = 1, 2,...,m, que interpolen en el intervalo $[x_k, x_{k+1}]$, y el cual cumpla con $x_k \le x^* \le x_{k+1}$, para calcular:

$$z^{**}_{x^*,j} = s_{kj}(x^*, y_j) . \tag{4.32}$$

c) Considerando las parejas de valores $s_i(y_j, z^{**}_{x^*,j})$, ajustar las funciones Spline respectivas.

d) Con $s_i(y_j, z^{**}_{x^*,j})$ apropiado calcular el valor z^*. Analíticamente esto es:

$$z^* = s_i(x^*, y^*) \quad , \quad y_i \le y^* \le y_{i+1} \quad ; \quad i = 1, 2, ..., m-1 . \tag{4.33}$$

Este procedimiento arrojaría resultados idénticos para z^*, si en el inciso (a), en lugar de seleccionar las parejas (x_i, z_{ij}), se hubiesen elegido las (y_i, z_{ij}), para cada x_i.

IV.2.5 SOLUCIÓN A LA INTERPOLACIÓN DEL FACTOR DE COMPRESIBILIDAD DE UN GAS NATURAL.

Para tal efecto, se empleará el método de interpolación SPLINE en tres dimensiones. Entonces, considerando los parámetros involucrados, es decir, "n" datos de presión, vector P(n); (abscisas), "m" de temperatura, vector T(m); (ordenadas) y "n × m" del factor de compresibilidad del gas matriz, Z(n,m); (cotas), se ejecutará el procedimiento siguiente:

1) Para cada T_j, j = 1, 2,...,m, seleccionar el conjunto de valores (p_i, Z_{ij}), i = 1, 2,...,n, y obtener las funciones Spline correspondientes, las cuales se valuarán en la abscisa de interés (presión a interpolar, Pint). En lenguaje *Visual Basic*, se tiene:

```
        ⋮
FOR J = 1 TO M
    FOR I = 1 TO N
        Z1(I)=Z(I,J)
    NEXT I
    CALL SPLINE (N, P(), Z1(), B(), C(), D())
    ZAUX (J) = SEVAL (N, Pint, P(), Z(), B(), C(), D())
NEXT J
        ⋮
```

Con lo cual tenemos el factor de compresibilidad ZAUX a Pint para todas las temperaturas.

2) Ahora, se llama a SPLINE para el conjunto de valores (T_j, $ZAUX_j$), j = 1, 2,...,m. Finalmente, las nuevas funciones generadas se valuarán en la ordenada de interés (temperatura a interpolar, Tint). En lenguaje *Visual Basic*, esto puede escribirse como:

```
        ⋮
CALL SPLINE (M, T(), ZAUX(), B(), C(), D())
Zint = SEVAL (M, Tint, T(), ZAUX(), B(), C(), D())
PRINT Zint
        ⋮
```

El programa de cómputo que aquí se presenta (Int3dim.exe, consultar archivo Indice.txt del anexo), se estructuró en lenguaje *Visual Basic*, y permite, además de interpolar el factor de compresibilidad de un gas natural a partir de su presión y temperatura, crear una base de

datos (archivos con extensión dat) que contenga la información básica del citado gas, evitándose con esto, el capturar tal información cada vez que se requiera efectuar alguna interpolación. En la Figura 4.4, se muestra de manera simplificada el diagrama de bloques del programa expuesto.

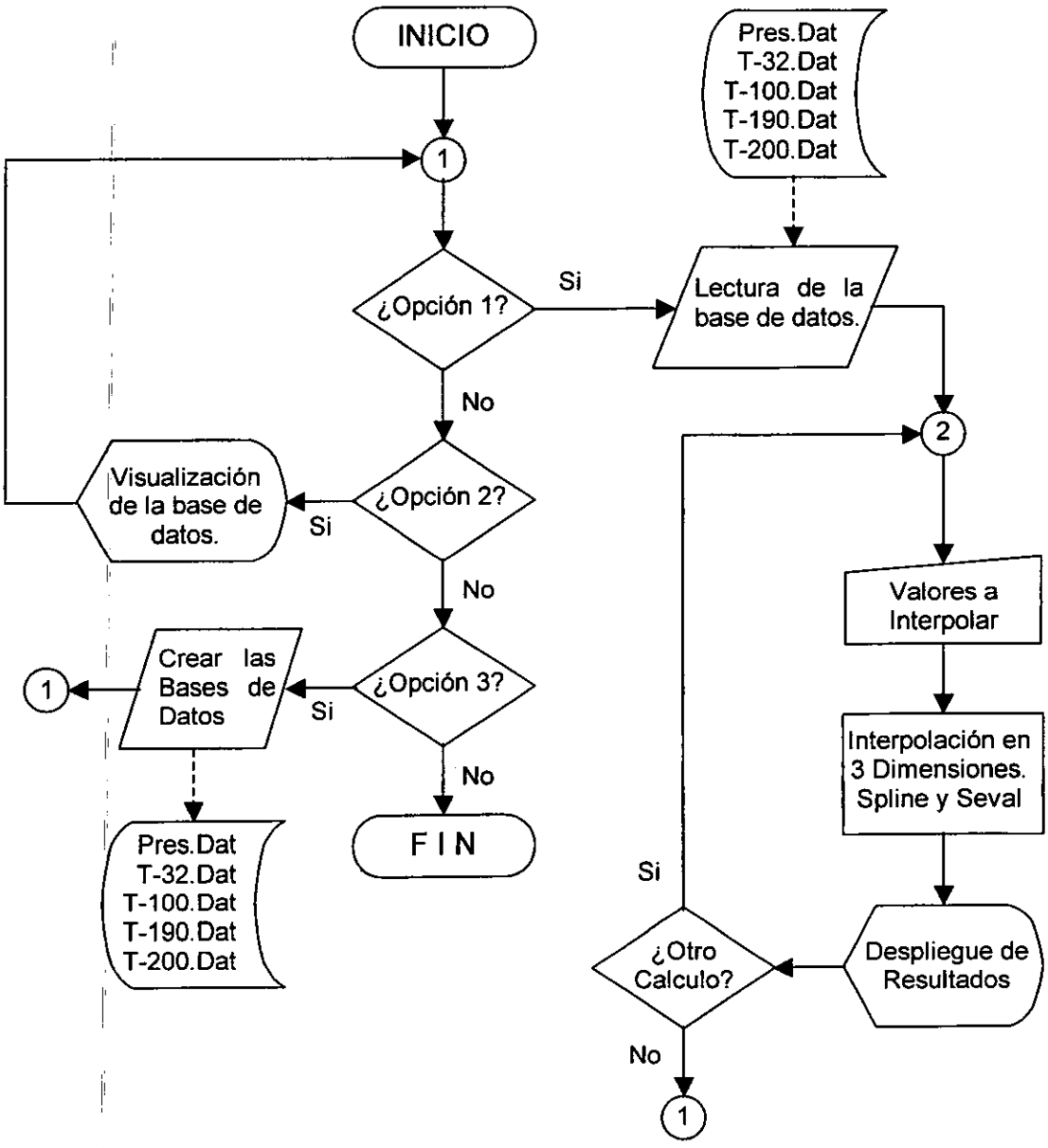


Figura 4.4. DIAGRAMA DE BLOQUES DEL PROGRAMA PARA INTERPOLACIÓN EN 3 DIMENSIONES

DATOS BÁSICOS PARA LA INTERPOLACIÓN.

Por un gasoducto fluye un gas natural que contiene 0.7% de nitrógeno. Fue analizado en laboratorio para obtener su factor de compresibilidad (z) para un rango de presión y temperatura de 1.4 a 5.0 lb/pg^2 y 32 a 280°F respectivamente y así poder determinar su comportamiento a través de la red de distribución. Si las condiciones de transporte se modificaran, los datos del análisis dejarían de ser válidos, es entonces, cuando la ayuda de un procedimiento de interpolación se hace necesaria. En la Tabla 4.1 se muestran los resultados del análisis.

Presión (lb/pg^2)	Factor de Compresibilidad (Z)			
	32 °F	100 °F	190 °F	280 °F
1.4	0.6885	0.8213	0.9097	0.9557
1.6	0.6593	0.8044	0.9009	0.9516
1.8	0.6433	0.7896	0.8943	0.9486
2.0	0.6369	0.7805	0.8899	0.9472
3.0	0.6981	0.7901	0.8932	0.9571
4.0	0.8103	0.8600	0.9356	0.9890
5.0	0.9333	0.9516	1.0050	1.0384

Tabla 4.1. DATOS DEL ANÁLISIS DE UN GAS NATURAL.

Para el programa de cómputo (Int3dim.exe) los rangos que se manejan son de 0.5 a 6 lb/pg^2 para la presión y de 0 a 300°F para la temperatura. Esta consideración está contemplada en el programa de cómputo. También, debe aclararse que en dicho programa el número de bases de datos se restringe a 4 para la temperatura (con sus factores de compresibilidad "z" correspondientes) y a una para la presión.

DISCUSIÓN DE RESULTADOS.

Los resultados obtenidos son aceptables, sin embargo, puesto que existe un rango muy extenso, mayor de 60°F, entre la información respectiva a cada temperatura, pueden presentarse resultados poco precisos. Al extrapolar, se tienen resultados *raros* o ilógicos, esto es debido a que SPLINE permite interpolaciones satisfactorias, mas no extrapolaciones. Por lo tanto, mientras mayor continuidad posea la información proporcionada, más acertada

será la interpolación ejecutada. Algunos resultados se muestran de manera tabular a continuación.

Presión	Factor de Compresibilidad (Z)		
(lb/pg² abs)	50 °F	150 °F	250 °F
1.0	0.8086	0.8961	0.9669
1.5	0.7173	0.8736	0.9379
2.5	0.6891	0.8436	0.9329
3.5	0.7681	0.8718	0.9558
4.5	0.8787	0.9398	1.0017
5.5	0.9844	1.0209	1.0697

Tabla 4.2. RESULTADOS DEL SIMULADOR DEL FACTOR DE COMPRESIBILIDAD.

IV.3 APROXIMACIÓN FUNCIONAL.

En las secciones anteriores se establecieron métodos de interpolación para encontrar la ecuación de la curva que contiene a todos y cada uno de los "n" puntos base. La curva continua de la Figura 4.5, representa la aproximación obtenida con algún método de interpolación. Ahora, se trata de encontrar la ecuación de una curva que, aunque no pase por todos los puntos, tenga pocas variaciones (sea suave, como la curva de trazo discontinuo de la Figura 4.5, página 117) y pase lo más cerca posible de todos ellos. Antes de aplicar cualquier técnica para tal problema, debe elegirse la forma de la curva que va a ser ajustada al conjunto de puntos dato. La ecuación de dicha curva puede obtenerse por conocimiento previo del problema, es decir, por la interpretación física del fenómeno, o en forma arbitraria, observando que ecuación conocida describe aproximadamente a esta curva.

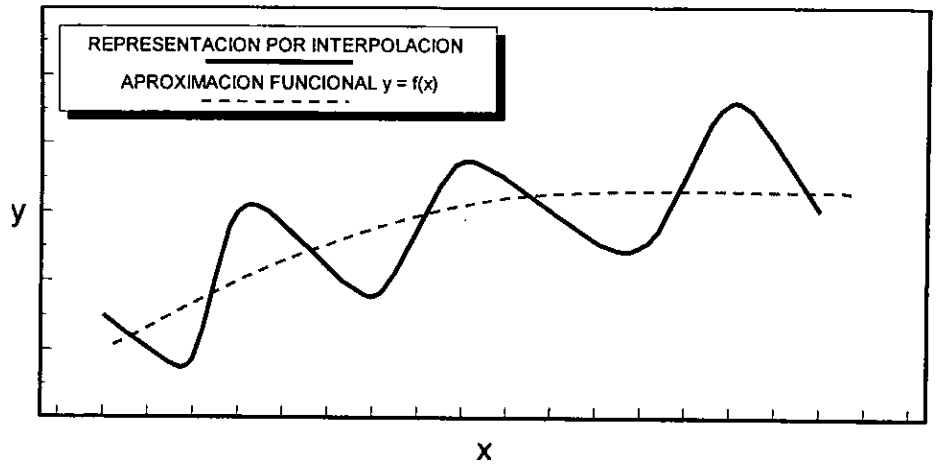


Figura 4.5. COMPARACIÓN GRÁFICA ENTRE LA INTERPOLACIÓN Y LA APROXIMACIÓN FUNCIONAL.

Las funciones de aproximación g(x) más comunes, son aquellas que involucran combinaciones lineales de funciones base o simples, $\{g_i(x)\}$, y tienen la forma:

$$g(x) = a_0 g_0(x) + a_1 g_1(x) + ... + a_n g_n(x) \ . \qquad (4.34)$$

Las funciones base más usadas son las monomiales $\{x^i\}$, i = 0, 1,2,...,n, las de Fourier {sin kx, cos kx}, k = 0, 1, 2,...,n, y las exponenciales $\{e^{bi}(x)\}$, i = 0, 1, 2,...,n.

Las combinaciones lineales de funciones monomiales desembocan en los polinomios de la forma:

$$f(x) \approx g(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n \ . \qquad (4.35)$$

La combinación lineal de funciones de Fourier conforma las aproximaciones siguientes:

$$f(x) \approx g(x) = a_0 + a_1 \cos x + a_2 \cos 2x + ... + a_n \cos nx + b_1 \operatorname{sen} x + b_2 \operatorname{sen} 2x + ... + b_n \operatorname{sen} nx \ . \qquad (4.36)$$

Las aproximaciones que emplean funciones exponenciales constituyen un modelo no lineal, o sea:

$$f(x) \approx g(x) = a_0 e^{b_0 x} + a_1 e^{b_1 x} + ... + a_n e^{b_n x} \quad . \tag{4.37}$$

Las aproximaciones racionales, aunque se emplean en menor grado, son:

$$f(x) \approx g(x) = \frac{a_0 + a_1 x + a_2 x^2 + ... + a_n x^n}{b_0 + b_1 x + b_2 x^2 + ... + b_m x^m} \quad . \tag{4.38}$$

En las técnicas que se describirán a continuación, sólo se considerará el caso donde el número "n" de puntos dato, sea mayor o igual, que el número "m" de coeficientes de la función de aproximación (grado del polinomio).

IV.3.1 MÉTODO DE LOS MÍNIMOS CUADRADOS[1].

Este método permite generar una función, que ajusta una curva suave a un conjunto de pares de puntos, tales que:

$$(x_1, y_1), (x_2, y_2), ..., (x_n, y_n) \quad , \quad x_1 < x_2 < ..... < x_n \quad ; \quad i = 1, 2, ..., n \quad .$$

Antes de mostrar las ecuaciones de esta técnica, debemos definir el concepto de residuo. Residuo es la diferencia de las ordenadas, entre la curva de ajuste y los datos base, para un valor de abscisa (x) dado (ver Figura 4.6, página 119). Representando como R_i a este residuo, analíticamente se tiene:

$$R_i = f(x_i) - y_i \quad , \quad i = 1, 2, ..., n \quad . \tag{4.39}$$

El método de los Mínimos Cuadrados consiste, entonces, en determinar los valores de los coeficientes de una función de aproximación, de grado "m", de manera que se minimice la suma de los cuadrados de los residuos. La forma de tal función es:

$$y = f(x) = a_0 + a_1 x + a_2 x^2 + ... + a_m x^m \quad . \tag{4.40}$$

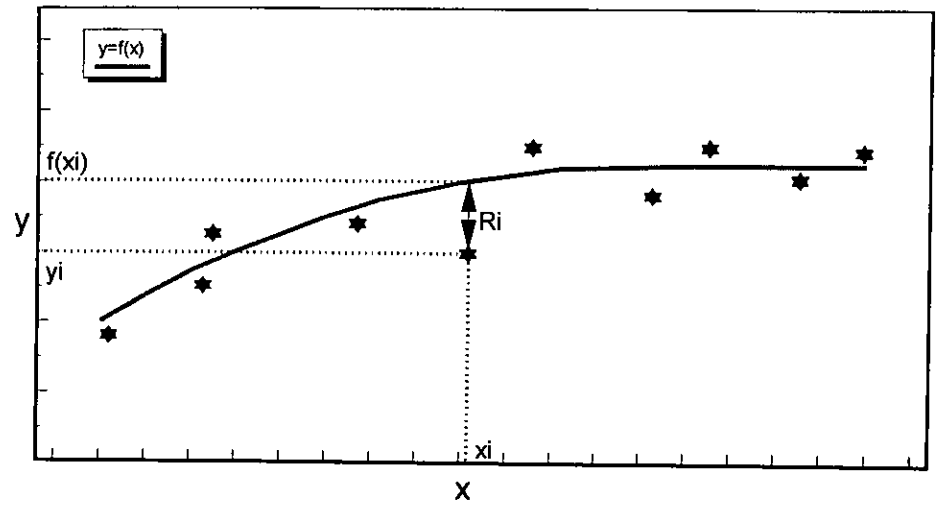


Figura 4.6. RESIDUOS (Ri) EN UN AJUSTE DE CURVAS.

El conjunto de ecuaciones a emplear, denominadas *normales*[10], es:

$$
\begin{aligned}
& na_0 + a_1\sum_{i=1}^{n}x_i + a_2\sum_{i=1}^{n}x_i^2 + \cdots + a_m\sum_{i=1}^{n}x_i^m = \sum_{i=1}^{n}y_i \\
& a_0\sum_{i=1}^{n}x_i + a_1\sum_{i=1}^{n}x_i^2 + a_2\sum_{i=1}^{n}x_i^3 + \cdots + a_m\sum_{i=1}^{n}x_i^{m+1} = \sum_{i=1}^{n}x_iy_i \\
& a_0\sum_{i=1}^{n}x_i^2 + a_1\sum_{i=1}^{n}x_i^3 + a_2\sum_{i=1}^{n}x_i^4 + \cdots + a_m\sum_{i=1}^{n}x_i^{m+2} = \sum_{i=1}^{n}x_i^2y_i \\
& \vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad\quad \vdots \qquad\quad \vdots \\
& a_0\sum_{i=1}^{n}x_i^m + a_1\sum_{i=1}^{n}x_i^{m+1} + a_2\sum_{i=1}^{n}x_i^{m+2} + \cdots + a_m\sum_{i=1}^{n}x_i^{m+m} \quad \sum_{i=1}^{n}x_i^my_i
\end{aligned}
\tag{4.41}
$$

En forma matricial este sistema puede ser escrito simplemente como:

$$\bar{Pc} = \bar{q}\ , \tag{4.42}$$

donde:

P = matriz de coeficientes del sistema (4.41) de orden (n × n).

$\bar{c}$ = vector de "m" coeficientes incógnita $\{a_i\}$, del sistema (4.41).

$\bar{q}$ = vector de "n" términos independientes del sistema (4.41).

El sistema $P\bar{c} = \bar{q}$, puede solucionarse empleando las subrutinas DECOMP y SOLVE. Pero, dado que la matriz P es simétrica, la memoria requerida puede reducirse a la mitad, además, P es una matriz positiva definida, por lo cual no es necesario buscar un elemento pivote, ya que los elementos de la diagonal principal serán todos diferentes de cero.

Regularmente, la matriz P tiene un número de condición (COND), demasiado grande. Esto provoca que cualquier error cometido en los datos, por ínfimo que éste sea, se traduzca en uno amplificado al calcular los coeficientes. De igual forma, cuando las funciones base $\{g_i(x)\}$, muestren dependencia, la matriz P será singular considerándose entonces, su número de condición como infinito.

Evidentemente, cualquier técnica que evite la generación de números de condición grandes en la matriz P, puede suponerse un buen detector de dependencia lineal entre las funciones base. En el siguiente subtema se describirá el método de la Descomposición del Valor Singular, que permite detectar y manejar el problema de la dependencia en las funciones base.

IV.3.2 DESCOMPOSICIÓN DEL VALOR SINGULAR (DVS) Y SUBRUTINA SVD[1].

El método de *Descomposición del Valor Singular (DVS)*, calcula los coeficientes del problema de Mínimos Cuadrados y está basado en la factorización de la matriz de esos coeficientes.

La técnica inicia con la conformación de la matriz de diseño a partir de los datos base, o sea, una matriz A de orden (m × n), cuyos elementos están definidos por la siguiente expresión:

$$a_{ij} = g_j(x_i)\,. \tag{4.43}$$

Si "y" denota al vector de "m" términos independientes $\{y_i\}$, y "c" al vector de "n" de componentes c_j, entonces la aproximación del modelo matemático lineal, está dada por:

$$\sum_{j=1}^{n} c_j g_j(x_i) \approx y_i \quad , \quad i = 1, 2, ..., m \ , \tag{4.44}$$

y en forma matricial se tiene:

$$A\bar{c} \approx \bar{y} \ . \tag{4.45}$$

El método DVS descompone a la matriz A en las matrices Σ, U y V. Σ es una matriz diagonal de orden (m × n) y sus elementos $\{\sigma_i\}$ no-negativos, son los valores singulares de la matriz A. Las matrices U, de orden (m × m), y V, de (n × n), son ortogonales y unitarias, y se emplean en la transformación del sistema $A\bar{c} \approx \bar{y}$, en un sistema equivalente $(\Sigma\bar{c} \approx \bar{y})$. Por lo que, si A se expresa como $U \Sigma V^t$, se tendrá que:

$$U \Sigma V^t \bar{c} \approx \bar{y} \ . \tag{4.46}$$

Dado el carácter ortogonal de U y V (esto es, $U U^t = I$ y $U^{-1} = U^t$), la expresión anterior puede escribirse como:

$$\Sigma V^t \bar{c} \approx U^t \bar{y} \ , \tag{4.47}$$

o también:

$$\Sigma \bar{c}' \approx \bar{y}' \ ,$$

donde:

$$\bar{c}' \approx V^t \bar{c} \ , \tag{4.49}$$

$$\bar{y}' \approx U^t \bar{y} \ . \tag{4.50}$$

Los valores singulares de la matriz A, están dados por las raíces cuadradas de los valores característicos de la matriz AA^t, que por cierto son iguales a los de la matriz A^tA. Los vectores singulares, izquierdo y derecho, son los vectores característicos de las matrices AA^t y A^tA, respectivamente, y constituyen las columnas de las matrices U y V.

Si las funciones base $g_j(x)$ fueran linealmente independientes en los punto dato, entonces los valores singulares serían diferentes de cero.

El método DVS debe considerar una tolerancia τ, la cual refleje la precisión de los datos originales. Cualquier valor singular σ_i mayor que dicha tolerancia, será aceptado y su correspondiente coeficiente c_j podrá evaluarse con la relación:

$$c_j = \frac{\bar{y}'_j}{\sigma_i} . \tag{4.51}$$

Si algún valor singular σ_i es menor que τ, se considerará nulo y su coeficiente asociado se igualará a cero. Una vez definidos los valores singulares máximo y mínimo, puede efectuarse el cociente entre ambos para obtener, de manera alternativa, el número de condición de la matriz A, es decir:

$$\text{Cond}(A) = \frac{\sigma_{máx}}{\sigma_{mín}} . \tag{4.52}$$

La técnica DVS se encuentra totalmente programada en la subrutina SVD (*Singular Value Decomposition*), la cual puede consultarse, junto con un programa fuente (Svd.bas), en el adjunto (consultar Apéndice A). Las siguientes sentencias en lenguaje *Visual Basic* permiten ilustrar el uso de la subrutina mencionada.

La matriz de diseño puede generarse con el siguiente grupo de instrucciones:

```
FOR I = 1 TO M
    T(I) = abscisa del i-ésimo punto base
    Y(I) = ordenada del i-ésimo punto base
    FOR J= 1 TO N
        A(I,J) = j-ésima función base evaluada en T(I)
NEXT J,I
```

En el caso de Aproximación Polinomial, una manera eficiente de constituir a la matriz A, es:

```
FOR I = 1 TO M
    T(I) = ...
    Y(I) = ...
    A(I,1) = 1.0
    FOR J= 2 TO N
        A(I,J) = T(I) * A(I,J-1)
NEXT J,I
```

A continuación, se debe incluir la llamada a SVD (leer los comentarios que aparecen en la subrutina para conocer los detalles de su uso), esto es:

```
CALL SVD (NM, M, N, A, SIGMA, 1, U, 1, V, IERR, WORK)
IF IERR <> 0 THEN
    Label.Caption = "ERROR DETECTADO POR SVD"
    END
ENDIF
```

El siguiente segmento detecta el valor singular máximo y mínimo y ubica los valores singulares despreciables, además, se inicializa el vector de coeficientes:

```
SIGMAMAX = 0.0
SIGMAMIN = SIGMA(1)
FOR J = 1 TO N
    IF SIGMA(J) > SIGMAMAX THEN SIGMAMAX = SIGMA(J)
    IF SIGMA(J) < SIGMAMIN THEN SIGMAMIN = SIGMA(J)
    C(J) = 0.0
NEXT J
COND = SIGMAMAX / SIGMAMIN
```

Ahora, se fija un error relativo de tolerancia, RELERR. Si por ejemplo, los datos tienen una exactitud de 3 decimales, entonces RELERR = 10^{-3}. Por lo tanto, la tolerancia (τ) en el error absoluto será:

```
TAU = RELERR * SIGMAMAX
```

El siguiente paso es obtener los coeficientes de la función de aproximación, en base a $\Sigma V^t \bar{c} \approx U^t \bar{y}$

```
FOR J = 1 TO N
    IF SIGMA(J) <= TAU THEN GOTO 60
    S = 0.0
    FOR I = 1 TO M
        S = S + U(I,J) * Y(I)
    NEXT I
    S = S / SIGMA(J)
    FOR I = 1 TO N
        C(I) = C(I) + S * V(I,J)
    NEXT I
60  NEXT J
```

Nótese, que M es el número de elementos de Y y el de renglones en A y U. N es el número de elementos de C y el de renglones en V. Todas la matrices tienen N columnas.

Los coeficientes están listos para ser utilizados:

```
FOR I = 1 TO N
    Label.Caption = "C(" & I & ") =" & C(I)
NEXT I
```

Para evaluar el modelo en cualquier punto TT, se propone:

```
YY = 0.0
FOR J = 1 TO N
    YY = YY + C(J) * (j-ésima función base evaluada en TT)
NEXT J
```

Para modelos polinomiales, se recomienda emplear el esquema de Horner:

```
YY = 0.0
FOR JB = 1 TO N
    J = N + 1 - JB
    YY = TT * YY + C(J)
NEXT I
```

La raíz cuadrada de la suma del cuadrado de los residuos, es la cantidad que se está minimizando y puede calcularse con:

```
RSQ = 0.0
FOR I = 1 TO M
    RI = 0.0
    FOR J = 1 TO N
        RI = RI + C(J) * A(I,J)
    NEXT J
    RSQ = RSQ + (RI - Y(I))^2
NEXT I
R = SQR(RSQ)
```

IV.3.3 AJUSTE DE FAMILIAS DE CURVAS.

Teniendo una función en tres dimensiones y con dos variables independientes, y = f(x, z), la cual a su vez conforma una familia de curvas como las mostradas en la Figura 4.7, puede ajustarse a un polinomio a través del procedimiento propuesto por el Dr. J. Tomas Limón. H. que se describe a continuación.

1) Ordenar de manera tabular los valores (x, y), para $z = z_i$, $i = 1, 2,...,n$.

2) Ajustar, mediante mínimos cuadrados (ver sección IV.3.1), un polinomio a estos puntos, obteniendo una expresión que de manera general tiene la forma:

$$y = a_{i,0} + a_{i,1}x + a_{i,2}x^2 + ... + a_{i,m}x^m \quad , \quad \text{para } z = z_i \quad ; \quad i = 1, 2,....., n \ , \tag{4.53}$$

donde "n" representa el número de datos y "m" el grado del polinomio. Los coeficientes $a_{i,0}$, $a_{i,1},...,a_{i,m}$, se determinan con el mismo ajuste polinomial.

3) El conjunto de ecuaciones del paso 2, puede reducirse a otro del tipo:

$$y = b_0 + b_1x + b_2x^2 + ... + b_mx^m \ , \tag{4.54}$$

donde los coeficientes b_i, son funciones de z, señalando que $a_{i,0}$, $a_{i,1},...,a_{i,m}$, corresponden a $z = z_i$. Por lo tanto, los coeficientes de la ecuación (4.54), están definidos por las expresiones siguientes:

$$\begin{aligned}
b_0 &= c_{0,0} + c_{0,1}z + c_{0,2}z^2 + \cdots + c_{0,m}z^m \\
b_1 &= c_{1,0} + c_{1,0}z + c_{1,1}z^2 + \cdots + c_{1,m}z^m \\
b_2 &= c_{2,0} + c_{2,0}z + c_{2,1}z^2 + \cdots + c_{2,m}z^m \\
&\vdots \quad\quad \vdots \quad\quad \vdots \quad\quad \vdots \quad\quad\quad \vdots \\
&\vdots \quad\quad \vdots \quad\quad \vdots \quad\quad \vdots \quad\quad\quad \vdots \\
b_m &= c_{m,0} + c_{m,1}z + c_{m,2}z^2 + \cdots + c_{m,m}z^m
\end{aligned} \tag{4.55}$$

Los coeficientes $c_{0,0}$, $c_{0,1}$, $c_{0,2},\ldots,c_{0,m}$, de la ecuación (4.55) para b_0, son el resultado de un ajuste polinomial en el que se han considerado las parejas de puntos $(z_1, a_{1,0})$, $(z_2, a_{2,0}),\ldots,(z_n, a_{n,0})$. De manera análoga, los coeficientes $c_{1,0}$, $c_{1,1}$, $c_{1,2},\ldots,c_{1,m}$, son el producto de un ajuste polinomial con los puntos $(z_1, a_{1,1})$, $(z_2, a_{2,1}),\ldots,(z_n, a_{n,1})$, y así sucesivamente, hasta calcular todos los coeficientes b_m restantes.

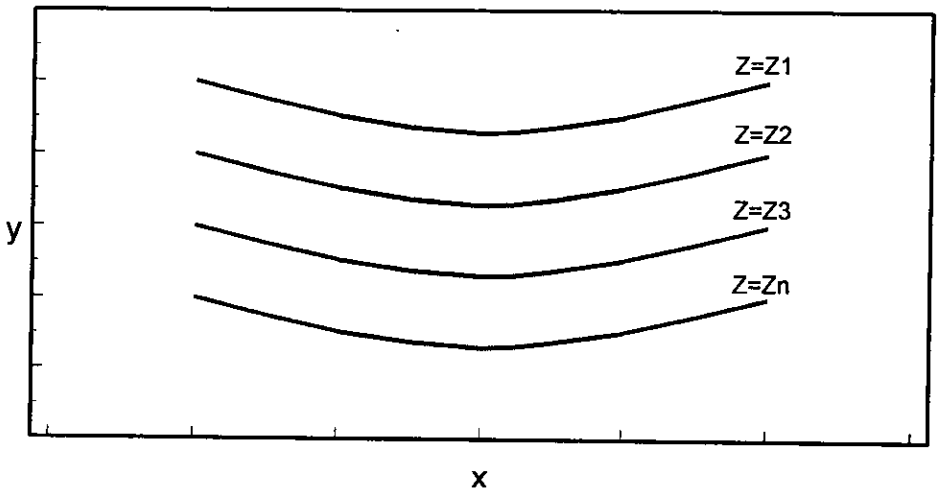


Figura 4.7. FAMILIA DE CURVAS EN 3 DIMENSIONES.

IV.3.4 INTRODUCCIÓN AL MÉTODO DE ESTIMACIÓN DE KRIGING.

En la exploración y explotación petrolera, normalmente se tiene un conjunto finito de valores espacialmente distribuidos de la variable en estudio, a partir de los cuales, debe reconstruirse el fenómeno con la fidelidad y confiabilidad suficiente para la toma de decisiones que coadyuven al desarrollo de un yacimiento. En la Industria Petrolera la obtención de información es difícil y costosa, por tanto, es necesario cuantificar con más exactitud y veracidad las variables recabadas, utilizando nuevas y mejores técnicas para el procesamiento de datos.

La Geoestadística, que fue definida por Georges Matheron (1962) como "la aplicación de las funciones aleatorias al reconocimiento y estimación de fenómenos naturales", considera que las variables de dichos fenómenos son de carácter mixto, es decir están compuestos de dos partes, una estructural y otra estocástica. La Figura 4.8 muestra una gráfica de mediciones

de porcentaje de mineral efectuadas a lo largo de cierta dirección en un yacimiento minero. Pueden apreciarse dos características: una local, de comportamiento errático o aleatorio, y otra general, con cualidades estructurales.

Describir fenómenos distribuidos espacialmente como lo son las formaciones geológicas (cimas y bases) o las propiedades físicas de una roca (porosidad, permeabilidad), permite evaluar las condiciones de saturación de agua, capacidad de flujo, índice de hidrocarburos, etcétera. Estas distribuciones pueden representarse por alguna función o modelo matemático, que resulta la mayoría de las veces tan complicado en su expresión como en su solución. Considérese que se desea obtener un plano configurado de las permeabilidades calculadas a través de pruebas de presión efectuadas en un cierto número de pozos (Figura 4.9). Lo primero que se observa es que la información no está distribuida regularmente en el espacio. Esto impediría la aplicación del método de las funciones Spline o del método de Mínimos Cuadrados. Otra característica importante, es que la información se encuentra agrupada en ciertas porciones del área en estudio, formando lo que se conoce como *nubes de información*. El ajuste de una superficie polinomial produciría, bajo estas circunstancias, resultados incoherentes debido a que la información más aislada estaría ejerciendo fuerte influencia sobre los coeficientes de los polinomios resultantes.

Por tales motivos, se han creado otros métodos de interpolación, tales como el de Ponderación con respecto al inverso de la distancia, Ponderación con respecto al inverso del cuadrado de la distancia, etcétera, los cuales empleados conjuntamente con la técnica de Búsqueda octal pueden producir resultados aceptables. Todos estos métodos, sin embargo, no pueden evitar el error inherente a todo proceso de interpolación. La Técnica de Estimación de Kriging (denominada así en honor del Doctor Daniel H. Krige de la escuela sudafricana), además de ser un método interpolador exacto y de minimizar el error, toma en cuenta las relaciones espaciales del fenómeno natural y permite la detección de rasgos característicos, tales como continuidad, variación en diferentes direcciones y la influencia de la variable alrededor de su vecindad. Contempla también, como ya se mencionó, que el valor de una variable es el resultado de 2 procesos, uno estructural y otro estocástico.

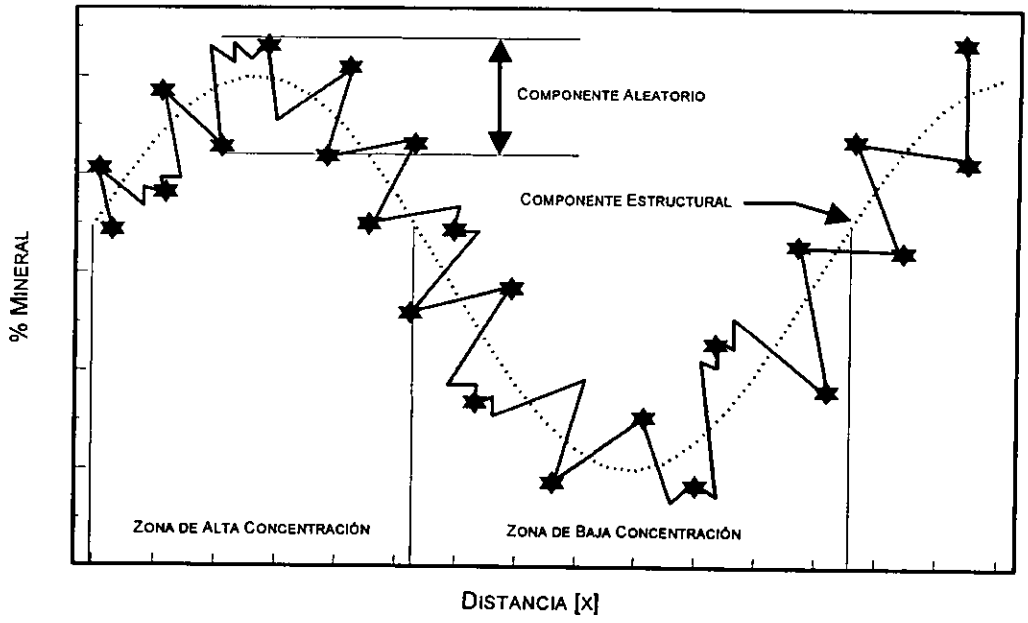


Figura 4.8. DISTRIBUCIÓN DE UNA VARIABLE NATURAL.

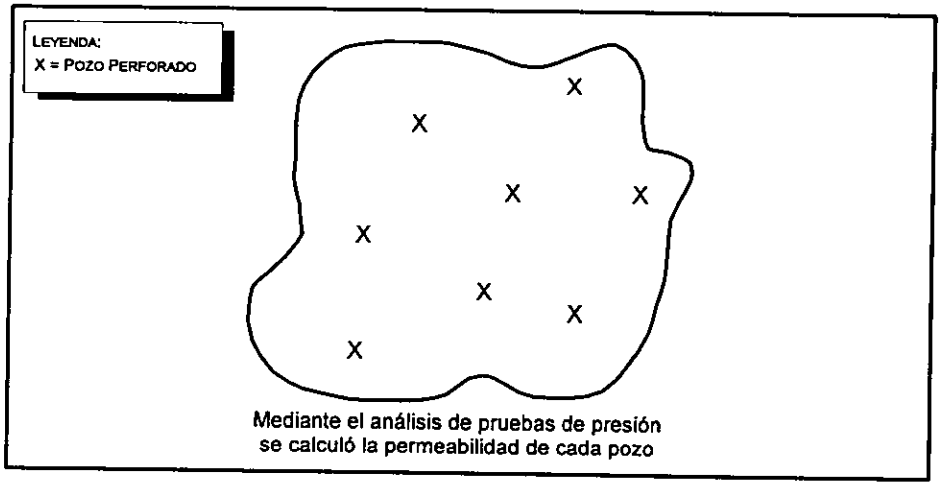


Figura 4.9. REPRESENTACIÓN GRÁFICA DE UN CAMPO PETROLERO.

El Método de Kriging es una técnica de estimación *local* la cual proporciona el mejor estimador lineal de las características desconocidas del fenómeno en estudio. El objetivo de la estimación local es encontrar el estimador más eficaz del valor medio de una variable asociada a un dominio limitado, y de dimensiones menores a las dimensiones de la zona de cuasi-estacionaridad del fenómeno, es decir, aquella zona en la cual la función acumulativa de distribución permanece constante bajo efectos de traslación, en otras palabras, las

distancias entre los puntos en estudio son menores o iguales a las de la zona considerada para propósitos de estimación.

La información requerida por este método consiste de un conjunto de datos (permeabilidad, porosidad, porcentaje de mineral, tiempos de reflexión, etcétera), e información estructural, es decir, el modelo del semivariograma que caracteriza la variabilidad en diferentes direcciones, de la zona estudiada. El semivariograma (γ), define el grado de continuidad de la variable, su zona de influencia y su mayor o menor variación en distintas direcciones. De la forma de la gráfica de espaciamiento entre pares de datos (h) vs. semivariograma $\gamma(h)$, se obtiene valiosa información acerca de una variable. Si se tienen puntos dispuestos regularmente a lo largo de una línea, el semivariograma puede calcularse, para incrementos de espaciamiento h, mediante la siguiente expresión:

$$\gamma(h) = \frac{1}{2} N \sum_{i=1}^{n} \left[Z(x_i + h) - Z(x_i) \right]^2 , \tag{4.56}$$

donde $Z(x_i)$ son los datos, x_i son las localizaciones tal que los datos estén disponibles en x_i y $x_i + h$, y N es el número de puntos.

La teoría desarrollada es como sigue: sea Z(x) una función aleatoria, con media E[Z(x)] = m, covariancia $E[Z(x+h)\,Z(x)] - m^2 = C(h)$ y variograma $E[\{Z(x+h) - Z(x)\}^2] = 2\gamma(h)$. El objetivo es estimar el valor medio de la variable $Z_v(x_0)$ asociada al dominio $V(x_0)$ con centro en el punto x_0. Los datos experimentales pueden estar dados por el conjunto de valores $\{Z_{v\alpha},\ \alpha = 1, 2,\ldots,n\}$, donde cada valor $Z_{v\alpha}$ está definido sobre el soporte $v\alpha$ con centro en el punto x_α. El valor $Z_v(x_0)$ será estimado linealmente, a partir de los n datos experimentales, con el estimador Z_k^*:

$$Z_k^* = \sum_{\alpha=1}^{n} \lambda_\alpha Z_{v\alpha} \ . \tag{4.57}$$

Los n coeficientes λ_α se calcularán asegurando que $\Sigma\lambda_\alpha = 1$, ya que así se garantiza que el valor esperado de Z_v sea igual al valor, también esperado, de Z_k^*, es decir:

$$E[Z_k^*] = E\left[\sum \lambda_\alpha Z_{v\alpha}\right] = m\sum\lambda_\alpha = m = E[Z_v]\,, \tag{4.58}$$

y la variancia de estimación σ_E^2 se obtiene con:

$$\sigma_E^2 = \bar{C}(V,V) - 2\sum_{\alpha=1}^{n}\lambda_\alpha \bar{C}(v_\alpha, V) + \sum_{\alpha=1}^{n}\sum_{\beta=1}^{n}\lambda_\alpha\lambda_\beta \bar{C}(v_\alpha, v_\beta)\,. \tag{4.59}$$

Aplicando el método de los Multiplicadores de Lagrange es posible encontrar el conjunto óptimo de coeficientes λ_α, y por supuesto sujetos a la condición $\Sigma\lambda_\alpha = 1$. Al igualar a cero las n derivadas parciales:

$$\frac{\partial}{\partial\alpha}\left[\sigma_E^2 - 2\mu\sum_{\alpha=1}^{n}\lambda_\alpha\right] = 0\,, \tag{4.60}$$

y al considerar la función restricción $\Sigma\lambda_\alpha = 1$, se define un sistema lineal de (n+1) ecuaciones y (n+1) incógnitas (los n coeficientes λ_α más el multiplicador de Lagrange μ), el cual se denomina *Sistema Kriging*, esto es:

$$\sum_{\beta=1}^{n}\lambda_\beta\gamma(v_\alpha, v_\beta) + \mu = \bar{\gamma}(v_\alpha, V) \quad , \quad \forall\,\alpha = 1, 2, \ldots\ldots, n\,, \tag{4.61}$$

$$\sum_{\beta=1}^{n}\lambda_\beta = 1\,. \tag{4.62}$$

Una vez resuelto el sistema para los coeficientes λ_α, la obtención de la variancia de estimación mínima σ_k^2 es inmediata:

$$\sigma_k^2 = \sum_{\alpha=1}^{n}\lambda_\alpha\bar{\gamma}(v_\alpha, V) + \mu - \gamma(V,V)\,. \tag{4.63}$$

El Sistema Kriging arrojará una solución única toda vez que la matriz de covariancia $\bar{C}(v_\alpha, v_\beta)$ sea una matriz definida positivamente. Dado que la técnica de Kriging minimiza el error del proceso de interpolación, permite la detección de rasgos característicos contemplando las relaciones espaciales del fenómeno natural en estudio, se le puede considerar como uno de los métodos de interpolación tridimensional más eficaces y confiables. A la fecha, se constituye como una herramienta de uso incipiente en el área de Ingeniería de Yacimientos Petroleros, con aplicaciones específicas en la caracterización de yacimientos.

IV.3.5 CÁLCULO DE LOS COEFICIENTES DEL MODELO PARA LA OPTIMIZACIÓN DE LA PERFORACIÓN.

El objetivo de este proyecto, es obtener los coeficientes de la expresión que define el comportamiento de la velocidad de penetración de la barrena sobre la formación, a partir de información del proceso mismo de perforación, como lo es: la densidad equivalente de circulación, la profundidad de perforación, el gradiente de presión, el peso sobre la barrena, la velocidad de la mesa rotaria, el desgaste de los dientes de la barrena y el número de Reynolds.

El término de *optimización de la perforación* se aplica a procedimientos que permiten seleccionar la hidráulica a chorro, el peso sobre la barrena, la velocidad de rotación, el tipo y las propiedades del lodo, el tipo de barrena, etcétera. El modelo que se adoptará es el propuesto por Bourgoyne y Young[11], el cual sólo permite la optimización del peso sobre la barrena, la velocidad de perforación y la hidráulica a chorro. La ecuación que simula tal modelo de perforación, es:

$$\mathrm{Ln}\left[\frac{dD}{dt}\right] = a_1 + \sum_{i=2}^{8} a_i x_i \ , \qquad (4.64)$$

donde:

$a_1,.....,a_8$ = coeficientes del modelo.

x_2 = 10,000 - D; efecto de compactación normal.

$x_3 = D^{0.69}(g_p - 9.0)$; efecto de sobre-compactación.

$x_4 = D(g_p - \rho_c)$; efecto de la presión diferencial.

$x_5 = Ln\left[\frac{\frac{w}{d} - \left(\frac{w}{d}\right)_t}{4.0 - \left(\frac{w}{d}\right)_t}\right]$; efecto del diámetro y el peso sobre la barrena.

$x_6 = Ln\left[\frac{N}{100}\right]$; efecto de la velocidad de rotación.

x_7 = -h; efecto del desgaste de los dientes de la barrena.

$x_8 = \frac{\rho q}{350\,\mu d_n}$; efecto hidráulico de la barrena (número de Reynolds).

D =profundidad [pie].

g_p = gradiente de presión [lb/gal].

ρ_c = densidad equivalente de circulación, [lb/gal].

d_n = diámetro de la tobera, [pg].

$\left(\frac{w}{d}\right)$ = peso sobre la barrena, [1,000 lb/pg]

$\left(\frac{w}{d}\right)_t$ = peso inicial sobre la barrena, [1,000 lb/pg].

$\left(\frac{dD}{dt}\right)$ = velocidad de penetración, [rpm/pies].

N = velocidad de la rotaria, [rpm].

h = desgaste del diente de la barrera, [fracción].

ρ = densidad del lodo, [lb/gal].

q = velocidad de flujo, [gal/min].

μ = viscosidad aparente de lodo a 10, 000 [seg^{-1}], [cp].

Para obtener el valor de los coeficientes del modelo es necesario emplear un procedimiento de aproximación funcional, por lo tanto, la técnica que determinará dichos coeficientes, será la de la Descomposición del Valor Singular (subrutina SVD), descrita en un subtema antecedente.

El programa desarrollado facilita la obtención de los 8 coeficientes del modelo, y además, la creación de archivos para la lectura/escritura de la información base (8 archivos con capacidad para 30 datos cada uno, extensión dto). El lenguaje de programación adoptado es el *Visual Basic*.

El programa de cómputo se presenta en el CD de consulta (Optperf.exe, ver Apéndice A), y su diagrama simplificado de bloques, en la Figura 4.10, página 136.

DATOS DEL PROBLEMA PLANTEADO.

Teóricamente, sólo se requieren 8 datos para resolver el modelo de perforación pero, como la ecuación no simula al 100% las condiciones de la misma, debe usarse un número razonable, que se propone sea cuando menos 30. Los datos, ver la Tabla 4.3, se tomaron del trabajo original de Bourgoyne y Young[11], y pertenecen a una formación de las costas de Louisiana, EE.UU.; (el peso inicial sobre la barrena es de 0.5 [1000 lb/pg]).

RESULTADOS.

Los coeficientes obtenidos al ejecutar el programa propuesto, son los siguientes:

$a_1 = 6.056605 \times 10^{-08}$ $\quad$ $a_5 = -3.05094 \times 10^{-08}$
$a_2 = 2.528517 \times 10^{-04}$ $\quad$ $a_6 = 2.038386 \times 10^{-08}$
$a_3 = 5.363132 \times 10^{-04}$ $\quad$ $a_7 = -2.954038 \times 10^{-08}$
$a_4 = -9.183395 \times 10^{-05}$ $\quad$ $a_8 = 2.285576 \times 10^{-08}$

La exactitud de los resultados depende estrechamente de la forma de la ecuación del modelo y de los valores de los parámetros de perforación usados, es decir, los valores de $x_2,...,x_8$. Además, la velocidad de penetración y de rotación, y el peso sobre barrena, deben registrarse en intervalos cortos para asegurar que son representativos del tipo de formación

atravesada. Es adecuado un intervalo de 2 a 5 pies, para garantizar el registro de datos representativos. Una vez conformado el polinomio del modelo, puede obtenerse, mediante ecuaciones matemáticas que quedan fuera del alcance de este proyecto, la velocidad de rotación, el peso sobre la barrena y la hidráulica óptimos, así como, los costos por pie perforado. La finalidad de optimizar tales parámetros es lograr maximizar el ritmo de penetración.

No. Dato	Profundidad, (D) [pie]	Vel. de Penet. (dD/dt) [pie/hr]	Peso sobre la Barrena, (w/d) [10,000 lb/pg]	Vel. de Rotación, (N) [rpm]	Desgaste del Diente, (h) [fracción]	No. de Reynolds	Densidad Equivalente, (ρc) [lb/gal]	Gradiente de Presión, (g_p) [lb/gal]
1	9515.0	23.0	2.58	113.0	0.77	0.964	9.5	9.0
2	9890.0	22.0	1.15	126.0	0.38	0.964	9.5	9.0
3	10130.0	14.0	0.81	129.0	0.74	0.827	9.6	9.0
4	10250.0	10.0	0.95	87.0	0.15	0.976	9.7	9.0
5	10390.0	16.0	1.02	78.0	0.24	0.984	9.7	9.0
6	10500.0	19.0	1.69	81.0	0.61	0.984	9.7	9.1
7	10575.0	13.0	1.56	81.0	0.73	0.984	9.7	9.2
8	10840.0	16.6	1.63	67.0	0.38	0.932	9.8	9.3
9	10960.0	15.9	1.83	65.0	0.57	0.878	9.8	9.4
10	11060.0	15.7	2.03	69.0	0.72	0.878	9.8	9.5
11	11475.0	14.0	1.69	77.0	0.20	0.887	10.3	9.5
12	11775.0	13.5	2.31	58.0	0.12	0.852	11.8	10.1
13	11940.0	6.2	2.26	67.0	0.20	0.976	15.3	12.4
14	12070.0	9.6	2.07	84.0	0.08	0.993	15.7	13.0
15	12315.0	15.5	3.11	69.0	0.40	1.185	16.3	14.4
16	12900.0	31.4	2.82	85.0	0.42	0.150	16.7	15.9
17	12975.0	42.7	3.48	77.0	0.17	1.221	16.7	16.1
18	13055.0	38.6	3.29	75.0	0.29	1.161	16.8	16.2
19	13250.0	43.4	2.82	76.0	0.43	1.161	16.8	16.2
20	13795.0	12.5	1.60	81.0	0.56	0.272	16.8	16.2
21	14010.0	21.1	1.04	75.0	0.46	0.201	16.8	16.2
22	14445.0	19.0	1.76	64.0	0.16	0.748	16.9	16.2
23	14695.0	18.7	2.00	76.0	0.27	0.819	17.1	16.2
24	14905.0	20.2	2.35	75.0	0.33	0.419	17.2	16.4
25	15950.0	27.1	2.12	85.0	0.31	1.290	17.0	16.5
26	15740.0	14.8	2.35	78.0	0.81	0.802	17.3	16.5
27	16155.0	12.6	2.47	80.0	0.12	0.670	17.9	16.5
28	16325.0	14.9	3.76	81.0	0.50	0.532	17.5	16.6
29	17060.0	13.8	3.76	65.0	0.91	0.748	17.6	16.6
30	20265.0	9.0	3.41	60.0	0.01	0.512	17.7	16.6

Tabla 4.3. DATOS DE PERFORACIÓN.

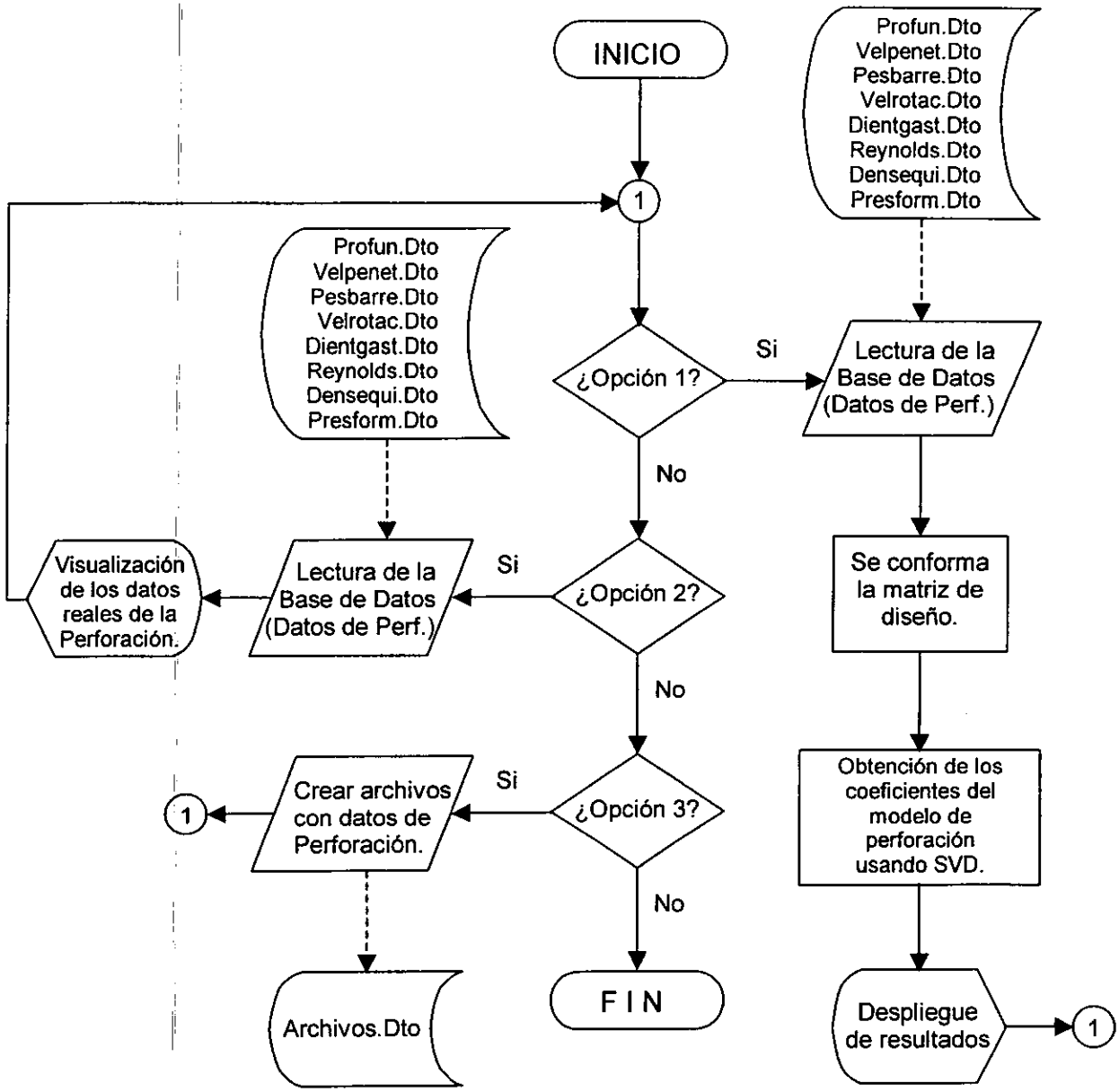


Figura 4.10. DIAGRAMA DE BLOQUES DEL PROGRAMA PARA OBTENER LOS COEFICIENTES DEL MODELO PARA LA OPTIMIZACIÓN DE LA PERFORACIÓN.

IV.4 PROBLEMAS RESUELTOS.

Problema Resuelto IV.1. Aplique el método de Kriging al conjunto de datos de porosidad mostrados en la Tabla 4.4. La gráfica del semivariograma promedio se muestra en la Figura 4.12. Esta información proviene de 65 pozos del Campo Miguel Alemán, situado en la porción sureste del Paleocanal de Chicontepec[16] (Figura 4.11).

Muestra	Coord. X [km]	Coord. Y [km]	Porosidad	Muestra	Coord. X [km]	Coord. Y [km]	Porosidad
1	143.34	76.40	0.198	34	138.37	79.38	0.16
2	143.12	76.78	0.19	35	138.76	79.34	0.24
3	142.70	76.68	0.12	36	139.16	79.31	0.11
4	142.74	77.50	0.16	37	139.48	79.38	0.15
5	142.56	77.80	0.13	38	139.90	79.36	0.247
6	143.06	77.92	0.13	39	138.59	79.74	0.21
7	140.52	77.12	0.17	40	138.98	79.67	0.11
8	140.74	77.42	0.16	41	138.16	79.71	0.24
9	140.74	77.54	0.21	42	139.97	79.70	0.21
10	141.52	77.46	0.16	43	139.77	79.74	0.198
11	141.24	77.96	0.13	44	139.98	78.64	0.19
12	141.80	77.92	0.17	45	140.14	79.00	0.166
13	140.20	78.30	0.198	46	141.15	79.48	0.16
14	140.60	78.36	0.14	47	139.89	80.62	0.17
15	140.94	78.20	0.23	48	139.12	80.70	0.21
16	138.72	78.18	0.15	49	140.32	74.03	0.15
17	139.00	78.34	0.185	50	141.14	79.06	0.20
18	139.38	78.36	0.14	51	141.32	79.95	0.15
19	138.16	78.84	0.17	52	140.63	79.59	0.14
20	138.76	78.70	0.17	53	140.60	79.23	0.13
21	138.38	78.48	0.14	54	140.88	85.12	0.198
22	139.16	78.66	0.17	55	141.71	78.51	0.17
23	139.58	78.72	0.17	56	141.35	78.52	0.15
24	140.38	78.64	0.14	57	141.85	78.88	0.13
25	140.84	78.74	0.13	58	142.37	78.89	0.23
26	141.32	78.54	0.14	59	141.54	79.60	0.16
27	140.46	78.93	0.2	60	141.57	79.67	0.185
28	140.12	79.68	0.13	61	141.72	79.25	0.26
29	140.33	79.36	0.16	62	141.35	79.26	0.13
30	140.59	79.60	0.26	63	142.16	79.20	0.15
31	140.46	79.92	0.17	64	142.60	77.85	0.13
32	138.97	79.00	0.15	65	141.85	77.91	17.00
33	138.76	78.97	0.175				

Tabla 4.4. DATOS DEL CAMPO MIGUEL ALEMÁN.

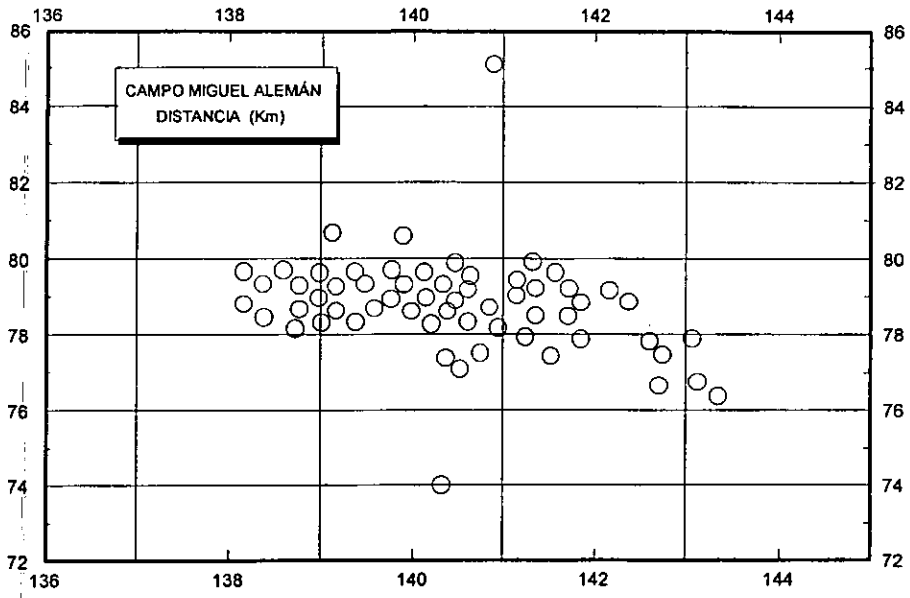


Figura 4.11. DISTRIBUCIÓN DE POZOS CON MUESTREO DE POROSIDAD.

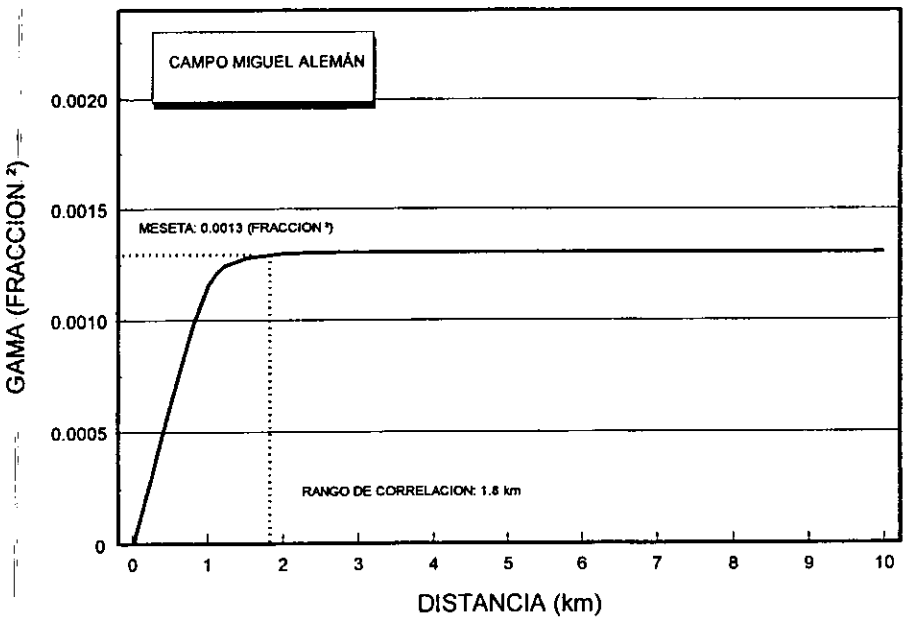


Figura 4.12. SEMIVARIOGRAMA PROMEDIO PARA LA POROSIDAD.

Solución:

El programa de cómputo empleado se estructuró en lenguaje *Visual Basic*, y mediante la técnica de Kriging estima el valor de una variable aleatoria natural, en este caso porosidad. Para fines prácticos, el yacimiento se subdivide en 12 bloques cuadrados de 2 km de ancho cada uno (4 celdas en la dirección X y 3 en la Y). Al *krigearse* cada celda se despliega una comparación, gráfica y numérica, entre el resultado obtenido y el calculado con métodos estadísticos convencionales.

Los resultados se observan en la Tabla 4.5 y se almacenan en un archivo llamado Resgloba.res, en general son congruentes, pero se ven afectados por el número de datos circunscritos a la celda en análisis en cada paso del proceso de estimación (celdas 1, 4, y 12, con 0, 1 y 2 datos respectivamente).

Para hacer más dinámico el uso de este programa, se incluye un archivo en código ASCII a manera de base de datos (Datos.reg, consultar Apéndice A), el cual contiene los datos requeridos por la técnica de Kriging. El diagrama simplificado de flujo se presenta en la página 140 (Figura 4.13).

	No. CELDA				
	3	6	9	12	
2	0,1777	0,1733	0,1700	0,1922	11
	0,162	0.163	0,206	0,181	
	0,000	0,000	0,133	0,172	
	1	4	7	10	

Tabla 4.5. ESTIMACIÓN DE LA POROSIDAD DEL CAMPO MIGUEL ALEMÁN.

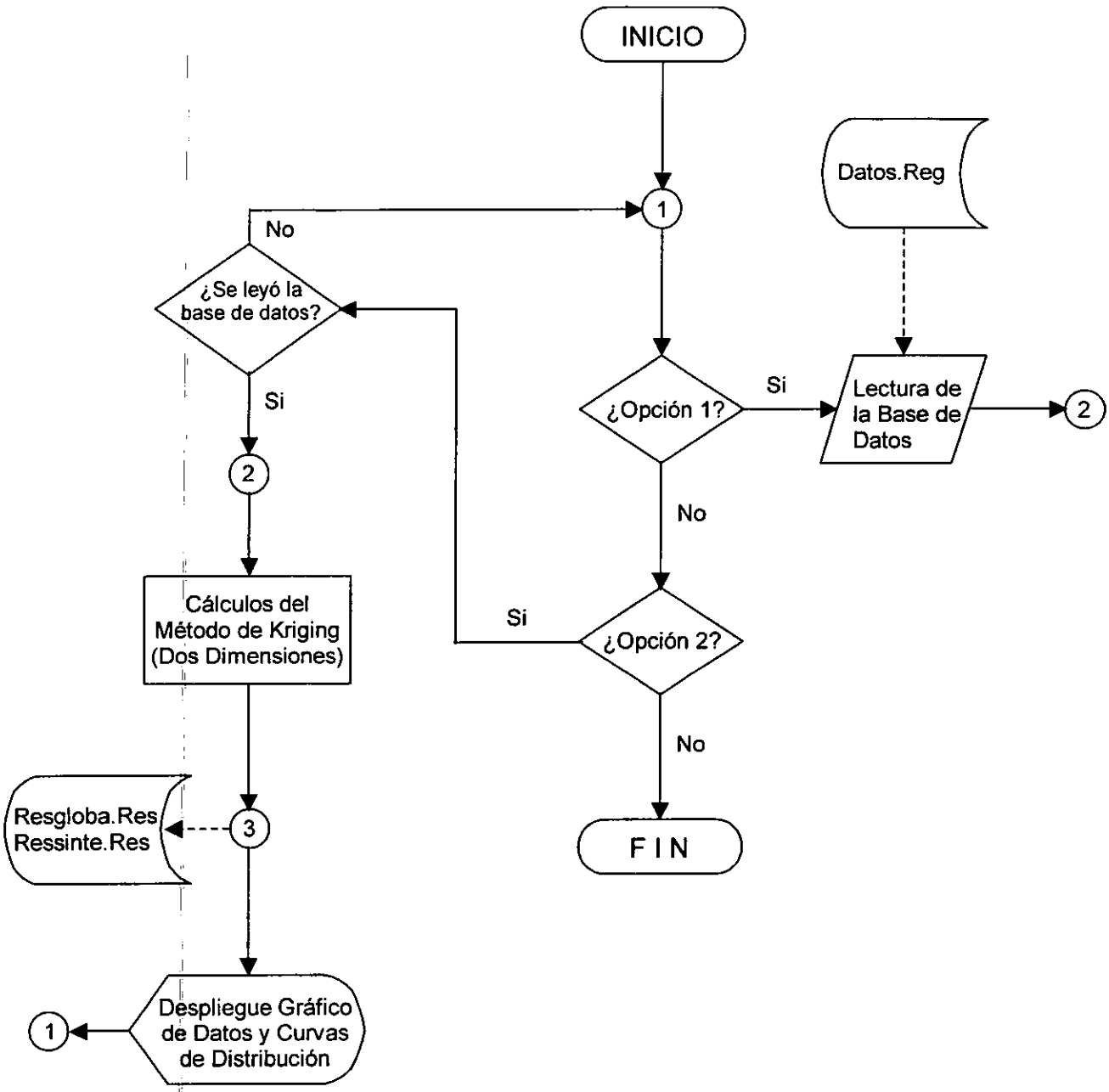


Figura 4.13. DIAGRAMA DE BLOQUES DEL PROGRAMA PARA EL MÉTODO DE KRIGING EN 2 DIMENSIONES.

IV.5 PROBLEMAS PROPUESTOS.

Problema Propuesto IV.1. Poettmann y Carpenter* han definido una expresión analítica que permite obtener la caída de presión en tuberías verticales con flujo multifásico, dicha expresión es la siguiente:

$$\frac{\Delta p}{\Delta h} = \frac{1}{144}\left[\rho + \frac{f\,(qM)^2}{2.979E-5\left(\rho\, d^5\right)}\right],$$

donde:

ρ = es la densidad de la mezcla sin resbalamiento, [lb_m/pie^3].

q = es el gasto de aceite, [bl/día].

M = es la masa asociada a un barril de aceite, [lb_m/bl_o a C.S.].

f = es el factor de fricción, el cual depende del producto qM.

d = es el diámetro interno de la tubería, [pg].

$\frac{\Delta p}{\Delta h}$ = es el gradiente de presión, [(lb/pg^2)/pie]

Encuentre, dado un cierto diámetro de tubería (d), el gasto de aceite por masa de la mezcla (qM) que produzca la mínima caída de presión. En base a la derivada de $\Delta p/\Delta h$ con respecto al gasto (qM) igualada a cero, se logra estimar el gasto que generará la caída mínima de presión. No obstante, el factor de fricción f depende también del gasto (qM), por lo que resulta necesario encontrar una función que exprese f en términos de (qM). Tal función deberá reemplazarse por f antes de proceder a la derivación. La Figura 4.14 expresa el factor de fricción f en términos de (qM)/d.

Emplee la técnica de los Mínimos Cuadrados o la de Descomposición del Valor Singular, para ajustar una función a las curvas mostradas (ver Figura 4.14). Sustitúyala en la ecuación de Poettmann y Carpenter, y defina la expresión del gasto (qM) en términos del diámetro d, para la cual la caída de presión en la tubería sea mínima.

*POETTMANN, F. H. Y CARPENTER, P. G., 'THE MULTIPHASE FLOW OF GAS, OIL AND WATER THROUGH VERTICAL FLOW STRINGS WITH APPLICATION TO THE DESIGN OF GAS LIFT INSTALLATIONS', DRILL AND PROD. PRAC., API, 1952.

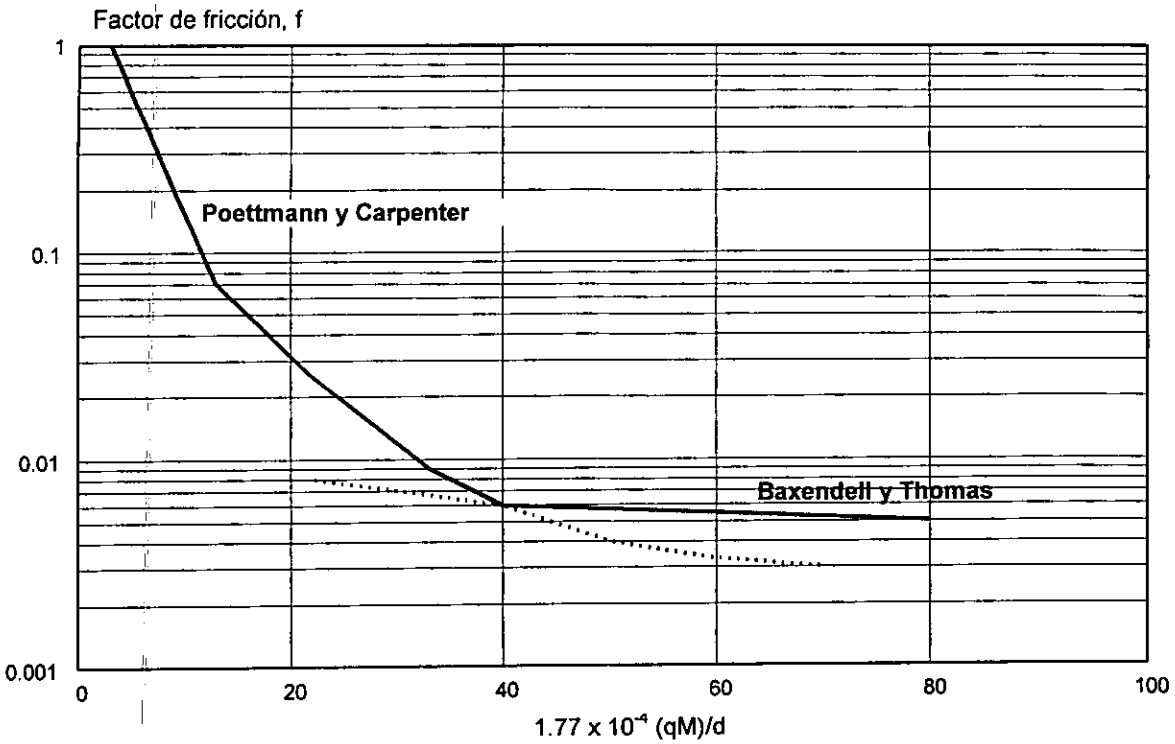


Figura 4.14. CORRELACIONES DEL FACTOR DE FRICCIÓN.
Correlaciones de Poettmann-Carpenter y Baxendell-Thomas.

INTEGRACIÓN NUMÉRICA | V

V.1 INTRODUCCIÓN.

Evaluar una integral definida

$$\int_a^b f(x)dx \ , \tag{5.1}$$

por métodos matemáticos, es frecuentemente una tarea difícil, aún cuando f(x) presente una forma analítica simple. Afortunadamente, el Análisis numérico ofrece diversas técnicas para integrar funciones definidas en forma tabular, obtenidas de algún experimento o simplemente valuando la función en cuestión. Una ventaja adicional de estos métodos numéricos, es que son fácilmente programables en cualquier equipo de cómputo, desde una calculadora de bolsillo hasta una computadora de gran capacidad.

En el área petrolera, es conveniente poseer técnicas sencillas y eficientes para dar solución a las integrales involucradas en el cálculo de algunos problemas, por ejemplo, puede citarse la Fugacidad o el Potencial de Gas Real, m(p).

La elección del método de integración más apropiado para un problema particular, está regida por la cantidad de información disponible sobre la función. Básicamente, en este capítulo se consideran sólo aquéllas funciones reales con una sola variable independiente, x, definidas en el intervalo [a, b].

Tales funciones pueden agruparse en 4 categorías:

1) Los valores de f(x), están disponibles sólo para ciertos puntos x_i del intervalo [a, b].
2) La función f(x), está bien definida y puede evaluarse para cualquier valor real, x, en el intervalo [a, b].
3) La definición de la función puede extenderse analíticamente al campo de los valores complejos de la variable x.
4) La función f(x) tiene una ecuación explícita disponible, con forma apropiada, para su manipulación simbólica.

Las funciones de la primera categoría son el resultado de mediciones experimentales para varios puntos x_i, los cuales a menudo no están uniformemente espaciados, o pueden haber sido obtenidos de tablas para valores equidistantes de x_i.

En las dos primeras categorías, se encuentran las funciones cuya diferenciación numérica es más difícil que la integración, esto ocurre puesto que la diferenciación numérica tiende a magnificar cualquier error inherente a los datos, mientras que la integración tiende a suavizar o disminuir el error. Si los valores de la función son conocidos o pueden calcularse con cierta exactitud, los métodos de integración basados en funciones Spline o funciones polinómicas presentan resultados satisfactorios, no obstante, si los valores de la función muestran *ruido*, entonces, los resultados pueden ser inexactos.

Como ejemplo de funciones en la tercera categoría, se encuentran las conformadas por complicadas expresiones trigonométricas o por funciones elementales; y si su extensión al dominio de los complejos es factible, la integración producirá resultados aceptables.

Para las funciones de la cuarta y última categoría, la diferenciación simbólica por computadora es más sencilla que la integración, ya que sólo involucra cálculos elementales.

En el ámbito de los Métodos numéricos, el término *regla* o *cuadratura*, engloba algoritmos con los cuales es posible calcular, aproximadamente, integrales definidas. En lo sucesivo, se detallarán solamente las cuadraturas, o reglas, aplicables a las funciones contempladas por las dos primeras categorías expuestas.

V.2 REGLAS DEL RECTÁNGULO Y DEL TRAPEZOIDE.

Suponga que [a, b] es un intervalo finito para la variable x, el cual está particionado en "n" subintervalos llamados paneles, $[x_i, x_{i+1}]$, i = 1, 2,...,n; además, $x_i = a$ y $x_{i+1} = b$, $x_1 < x_2 < \ldots < x_{n+1}$. Definiendo, el ancho de cada panel:

$$h_i = x_{i+1} - x_i \ . \tag{5.2}$$

Sea f(x) una función definida en el mismo intervalo [a, b] y supóngase que se desea una aproximación de la integral:

$$I(f) = \int_a^b f(x)dx \ . \tag{5.3}$$

Sencillamente, I(f) puede expresarse como la suma de las integrales sobre cada panel h_i (Figura 5.1), o sea:

$$I(f) = \sum_{i=1}^{n} I_i \ , \tag{5.4}$$

donde:

$$I_i = I_i(f) = \int_{x_i}^{x_{i+1}} f(x)dx \ . \tag{5.5}$$

Una *regla de cuadratura simple* es una fórmula que permite aproximar, de manera individual, cada I_i. *Una regla de cuadratura compuesta*, es una fórmula que calcula aproximadamente la integral I(f), mediante la suma de las reglas de cuadratura simples de cada subintegral, I_i.

Las reglas de cuadratura simple más usadas son la del Rectángulo y la del Trapezoide. La Regla de Rectángulo valúa la función f(x) en los puntos medios de los paneles, esto es, en el punto:

$$y_i = \frac{x_i + x_{i+1}}{2} \quad , \quad i = 1, 2, ..., n \ . \tag{5.6}$$

Y cada integral I_i, se obtiene como el área de un rectángulo de base h_i, y de altura $f(y_i)$, por lo que:

$$I_i \approx h_i f(y_i) \ . \tag{5.7}$$

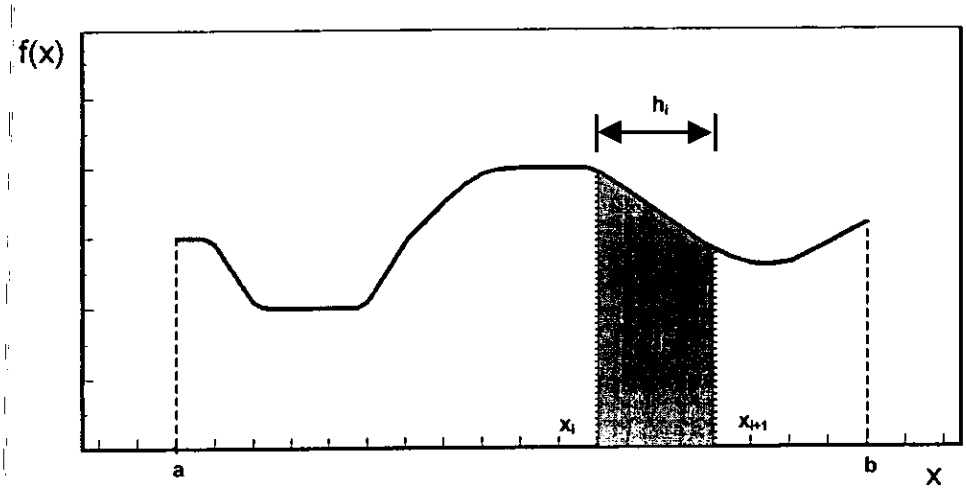


Figura 5.1. ANCHO DEL PANEL h_i.

En base a lo anterior, la Regla Compuesta del Rectángulo será la siguiente (ver Figura 5.2):

$$R(f) = \sum_{i=1}^{n} h_i f(y_i) \ . \tag{5.8}$$

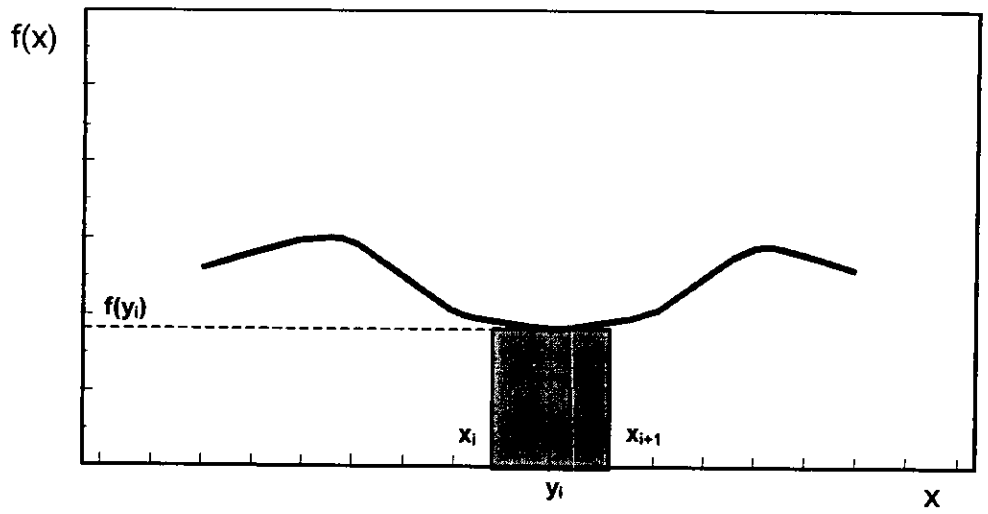


Figura 5.2. REGLA DEL RECTÁNGULO.

Por otro lado, la Regla del Trapezoide valúa la función en los puntos extremo de cada panel. La integral I_i, es aproximada por el área del trapecio con base h_i, cuya altura varía, de izquierda a derecha, o viceversa, de $f(x_i)$ a $f(x_{i+1})$ (lo que puede apreciarse en la Figura 5.3). Por lo tanto:

$$I_i(f) \approx h_i \frac{f(x_i)+f(x_{i+1})}{2}. \tag{5.9}$$

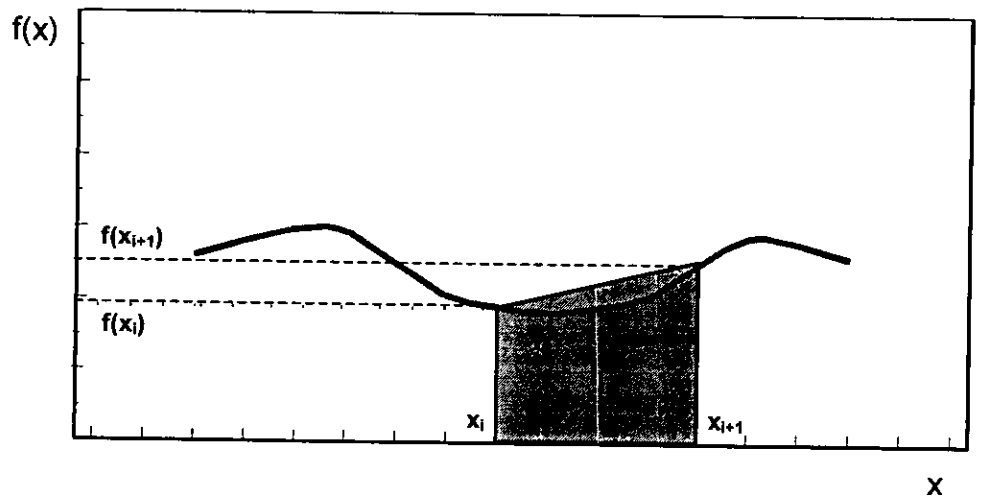


Figura 5.3. REGLA DEL TRAPEZOIDE.

Y la Regla Compuesta del Trapezoide, estará dada por la siguiente expresión:

$$T(f)=\sum_{i=1}^{n} h_i \frac{f(x_i)+f(x_{i+1})}{2} . \tag{5.10}$$

Si f(x) es una función continua –o simplemente integrable según Riemann– en el intervalo [a, b], y si $h = x_i$ entonces, ambas cuadraturas convergen al resultado exacto conforme el ancho de cada subintervalo decrece[1], es decir:

$$\lim_{h\to 0} R(f)=I(f) , \tag{5.11}$$

$$\lim_{h\to 0} T(f)=I(f) . \tag{5.12}$$

Ahora, surge la interrogante: ¿Qué tan rápido convergen estas reglas? La Cuadratura del Rectángulo emplea una interpolación constante (grado cero), en cada subintervalo; y la del Trapezoide, una interpolación lineal (grado uno). Por sentido común, podría esperarse mayor exactitud en la Regla Trapezoidal.

Sean [a, b] = [0, 1], n = 1 (un sólo panel) y la función f(x) = x, tal como se muestra en las Figuras 5.4 y 5.5. La Regla Trapezoidal es evidente que no generará error, ya que la interpolación lineal concuerda para cualquier punto con la función f(x). Aún así, la Cuadratura del Rectángulo brinda un resultado también libre de error, a pesar de que la función interpoladora no coincide con la función f(x) en el punto x = ½. El error promedio es cero en ambos casos.

¿El ejemplo anterior es típico o simplemente es *suerte* de la Cuadratura del Rectángulo? ¿En general, qué regla es más exacta? Para responder estas inquietudes, se realizará un análisis basado en suposiciones sobre una función cualquiera.

Considere una función f(x), que posee 5 derivadas continuas, cuyos valores no son muy grandes. También, considere el panel $[x_i, x_{i+1}]$. La expansión de f(x) con respecto al punto y_i, localizado en el centro del panel, es:

$$f(x) = f(y_i) + (x - y_i)f'(y_i) + \frac{1}{2}(x - y_i)^2 f''(y_i) + \frac{1}{6}(x - y_i)^3 f'''(y_i) + \frac{1}{24}(x - y_i)^4 f^{iv}(y_i) + \ldots\ldots \quad (5.13)$$

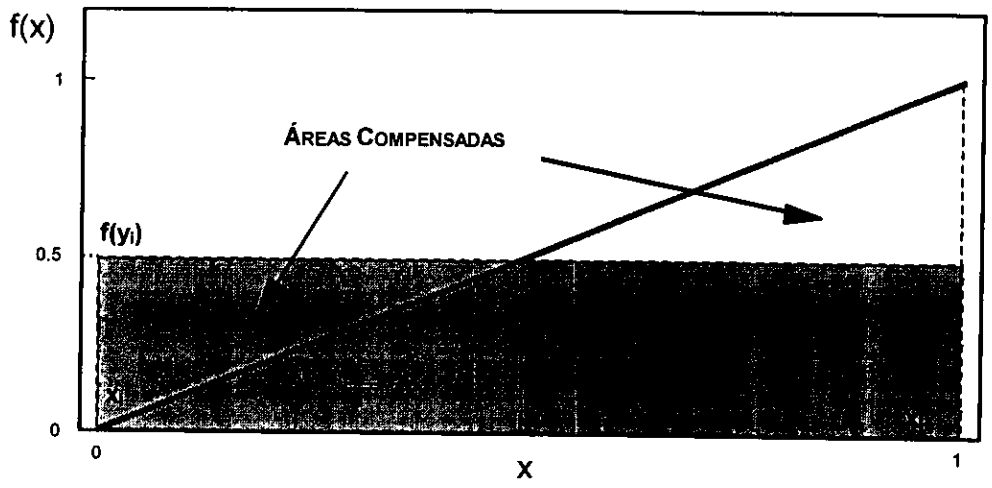


Figura 5.4. REGLA DEL RECTÁNGULO. FUNCIÓN f(x) = x EN EL INTERVALO [0,1].

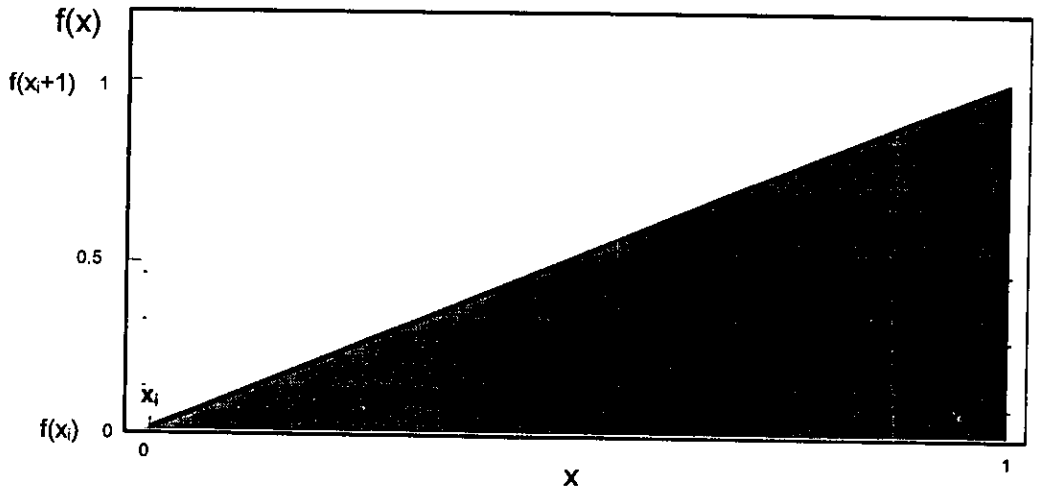


Figura 5.5. REGLA DEL TRAPEZOIDE. FUNCIÓN f(x) = x EN EL INTERVALO [0,1].

La suposición considerada es que los términos denotados por el símbolo "....", son despreciables, es decir, menos significativos que los mostrados explícitamente.

Al integrar cada uno de los términos de la serie anterior desde x_i a x_{i+1}, se observa lo siguiente:

$$\int_{x_i}^{x_{i+1}}(x-y_i)^p\,dx=\begin{cases} h_i \Rightarrow p=0 \\ 0 \Rightarrow p=1 \\ \left(\frac{h_i^3}{12}\right) \Rightarrow p=2 \\ 0 \Rightarrow p=3 \\ \left(\frac{h_i^5}{80}\right) \Rightarrow p=4 \end{cases}, \tag{5.14}$$

nótese, que las potencias impares al ser integradas son cero; consecuentemente:

$$\int_{x_i}^{x_{i+1}} f(x)dx = \underbrace{h_i f(y_i)}_{\substack{\text{Cuadratura del} \\ \text{Rectángulo}}} + \frac{1}{24}h_i^3 f''(y_i)+\frac{1}{1920}h_i^5 f^{iv}(y_i)+\ldots\ldots . \tag{5.15}$$

La expresión anterior indica que para valores pequeños de h_i, el error para cada panel, en la aproximación por la Cuadratura del Rectángulo, es del orden de $\frac{1}{24}h_i^3 f''(y_i)$, más otros términos poco significativos.

Retomando una expansión en serie de Taylor y sustituyendo $x = x_i$ y $x = x_{i+1}$, se obtiene:

$$f(x_i)=f(y_i)-\frac{1}{2}h_i f'(y_i)+\frac{1}{8}h_i^2 f''(y_i)-\frac{1}{48}h_i^3 f'''(y_i)+\frac{1}{384}h_i^4 f^{iv}(y_i)+\ldots\ldots , \tag{5.16}$$

$$f(x_{i+1})=f(y_i)+\frac{1}{2}h_i f'(y_i)+\frac{1}{8}h_i^2 f''(y_i)+\frac{1}{48}h_i^3 f'''(y_i)+\frac{1}{384}h_i^4 f^{iv}(y_i)+\ldots\ldots , \tag{5.17}$$

entonces, sumando y reordenando las expresiones anteriores:

$$\frac{f(x_i)+f(x_{i+1})}{2}=f(y_i)+\frac{1}{8}h_i^2 f''(y_i)+\frac{1}{384}h_i^4 f^{iv}(y_i)+\ldots\ldots . \tag{5.18}$$

Combinando esta última expresión con la de la integral que representa el error de la Cuadratura del Rectángulo, se llega a:

$$\int_{x_i}^{x_{i+1}} f(x)dx = h_i \frac{f(x_i)+f(x_{i+1})}{2} - \frac{1}{12}h_i^3 f''(y_i) - \frac{1}{480}h_i^5 f^{iv}(y_i) + \ldots\ldots \quad . \tag{5.19}$$

Puede observarse que el error para cada panel en la Regla del Trapezoide, cuando h_i es pequeña, es $-\frac{1}{12}h_i^3 f''(y_i)$, más otros términos menos significativos.

El error total para cada regla será la sumatoria del error presente en cada panel. Si se considera la siguiente convención:

$$E = \frac{1}{24}\sum_{i=1}^{n} h_i^3 f''(y_i) \quad , \tag{5.20}$$

$$F = \frac{1}{1920}\sum_{i=1}^{n} h_i^5 f^{iv}(y_i) \, , \tag{5.21}$$

el error total resulta:

$$I(f) = R(f) + E + F + \ldots\ldots \qquad (\text{Rectángulo}) \, , \tag{5.22}$$

$$I(f) = T(f) - 2E - 4F + \ldots\ldots \qquad (\text{Trapezoide}) \, . \tag{5.23}$$

Si h_i es suficientemente pequeña, entonces $h_i^5 << h_i^3$, y si f^{iv} no presenta un comportamiento errático, entonces F << E.

Las conclusiones derivadas del análisis anterior son:

- Para la gran mayoría de las funciones f(x), la Regla del Rectángulo es cerca de 2 veces más exacta que la del Trapezoide.
- La diferencia entre los valores obtenidos con cada regla puede usarse para estimar el error en la integración. Esto es:

$$R(f) + E + F = T(f) - 2E - 4F \, , \tag{5.24}$$

$$R(f) - T(f) = -3E - 5F \approx -3E \, . \tag{5.25}$$

- La diferencia entre el resultado arrojado por el uso de una regla, antes y después de duplicar el número paneles, también ofrece una aproximación del error en la integración, pues al duplicarse el número de paneles se cuadruplica la exactitud de la cuadratura.

La técnica de aumentar repetidamente la cantidad de paneles al doble y de calcular el error, es susceptible de ser programada para generar un método que automáticamente determine el número de paneles necesarios, logrando que el valor de la integral aproximada esté dentro de una tolerancia de error preestablecida. Este método se aplica en cuadraturas más sofisticadas, que se expondrán, en detalle, más adelante (sección V.5).

V.3 CUADRATURA SPLINE[1].

La interpolación cúbica Spline permite obtener una fórmula sencilla de cuadratura. Para $x_i \le x \le x_{i+1}$, la función Spline que interpola a todos los puntos base de f(x), es:

$$s(x) = f_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \ , \tag{5.26}$$

entonces para $a = x_1$, $b = x_{n+1}$ y $h = x_{i+1} - x_i$, siendo "n" el número de paneles, se define la Cuadratura Compuesta Spline como:

$$\int_a^b f(x)dx \approx \int_a^b s(x)dx = \sum_{i=1}^{n}\left[h_i f_i + \frac{1}{2}h_i^2 b_i + \frac{1}{3}h_i^3 c_i + \frac{1}{4}h_i^4 d_i \right] . \tag{5.27}$$

Los coeficientes b_i, c_i y d_i se calculan con la subrutina SPLINE (descrita en el Capítulo IV). Como se tienen "n+1" nodos o puntos base, SPLINE debe invocarse con N = n+1.

La función s(x) puede ser representada también como (consultar sección IV.2.4):

$$s(x) = wf_{i+1} + \overline{w}f_i + h_i^2\left[(w^3 - w)\sigma_{i+1} + (\overline{w}^3 - \overline{w})\sigma_i \right] , \tag{5.28}$$

donde:

$$w = 1 - \overline{w} = \frac{(x - x_i)}{h_i} \quad , \tag{5.29}$$

$$\sigma_i = \frac{s''(x_i)}{6} = \frac{c_i}{3} \quad . \tag{5.30}$$

Las σ_i son la solución del sistema tridiagonal que se conforma con las ecuaciones anteriores, y que se discutió en la sección IV.2.4. Continuando con el desarrollo, se observa que:

$$\int_{x_i}^{x_{i+1}} s(x)dx = h_i \int_0^1 s(w)dw \quad , \tag{5.31}$$

además:

$$\int_0^1 w\, dw = \frac{1}{2} \quad , \tag{5.32}$$

$$\int_0^1 (w^3 - w)dw = -\frac{1}{4} \quad , \tag{5.33}$$

en consecuencia:

$$\int_{x_i}^{x_{i+1}} s(x)dx = h_i \frac{f_i + f_{i+1}}{2} - h_i^3 \frac{\sigma_i + \sigma_{i+1}}{4} \quad . \tag{5.34}$$

Es evidente que la fórmula de Cuadratura Spline es igual a la Trapezoidal, más un término de corrección que involucra a σ_i. Una ecuación más simple, para calcular la Cuadratura Compuesta Spline es:

$$\int_a^b s(x)dx = \sum_{i=1}^{n} \left[h_i \frac{f_i - f_{i+1}}{2} - h_i^3 \frac{c_i - c_{i+1}}{12} \right] . \tag{5.35}$$

Por lo tanto, sólo los vectores de datos (x, f) y el de segundas derivadas (c), obtenido por SPLINE, se requirieren para evaluar la integral. La exactitud de la Cuadratura Spline puede calcularse mediante la siguiente expresión:

$$h_i^3 \frac{c_i + c_{i+1}}{12} = h_i^3 \frac{s''(x_i) + s''(x_{i+1})}{24} \approx \frac{h_i}{12} f''(y_i) \quad . \tag{5.36}$$

Se aprecia entonces, que el término de corrección provisto por la Cuadratura Spline aproxima el error de la Regla Trapezoidal.

En la literatura no se han encontrado resultados prácticos sobre la utilización de la Cuadratura Spline, pero en base a este análisis, parece revestir gran utilidad; aunque se recomienda para casos en los cuales los puntos dato son fijos y los métodos numéricos, que se describirán más adelante, no están disponibles.

V.4 REGLA DE SIMPSON.

En la sección V.2, se expusieron las Cuadraturas del Rectángulo y del Trapezoide, éstas son:

$$R(f) = \sum_{i=1}^{n} h_i f(y_i) \quad , \tag{5.37}$$

$$y_i = \frac{x_i + x_{i+1}}{2} \quad , \tag{5.38}$$

$$T(f) = \sum_{i=1}^{n} \frac{f(x_i) + f(x_{i+1})}{2} \quad , \tag{5.39}$$

además, se probó que los errores de esas reglas están dados por:

$$I(f) - R(f) = E + F + \ldots\ldots \quad , \tag{5.40}$$

$$I(f) - T(f) = 2E - 4F + \ldots\ldots \quad , \tag{5.41}$$

donde:

$$E = \frac{1}{24}\sum_{i=1}^{n} h_i^3 f''(y_i) \quad , \tag{5.42}$$

$$F = \frac{1}{1920}\sum_{i=1}^{n} h_i^5 f^{iv}(y_i) \quad . \tag{5.43}$$

Combinando apropiadamente ambas reglas es factible crear una nueva cuadratura cuya fórmula de error no contenga los términos de E. Como R(f) es cerca de 2 veces más exacta que T(f), la combinación idónea es[1]:

$$S(f) = \frac{2}{3}R(f) + \frac{1}{3}T(f) \quad , \tag{5.44}$$

y desarrollando la expresión anterior, se llega a:

$$S(f) = \sum_{i=1}^{n} \frac{1}{6} h_i \left[f(x_i) + 4f\left(\frac{x_i + x_{i+1}}{2} \right) + f(x_{i+1}) \right] \quad . \tag{5.45}$$

El lector identificará la ecuación (5.45) como la Cuadratura Compuesta de Simpson, cuyo error puede obtenerse directamente de las fórmulas para los errores en las reglas del Trapezoide y del Rectángulo, o sea:

$$\begin{aligned} I(f) - S(f) &= \frac{2}{3}[I(f) - R(f)] + \frac{1}{3}[I(f) - T(f)] \\ &= \frac{2}{3}(E + F + ...) + \frac{1}{3}(-2E - 4F + ...) \\ &= \left(\frac{2}{3} - \frac{2}{3} \right) E + \left(\frac{2}{3} - \frac{4}{3} \right) F + ... \\ &= -\frac{2}{3}F + ... \\ &= -\frac{1}{2280}\sum_{i=1}^{n} h_i^5 f^{iv}(y_i) + ... \end{aligned} \quad . \tag{5.46}$$

Obsérvese que S(f) está basada en una interpolación de segundo grado y que su término de error involucra derivadas de cuarto orden, por lo cual, la Cuadratura de Simpson es exacta para funciones cúbicas.

Si la longitud de cada panel es bipartida, entonces cada término h_i^5 en la fórmula del error, decrece a razón de $^1/_{32}$. El número total de términos se incrementa al doble, por lo consiguiente, la reducción del error total es de cerca de $^1/_{16}$. Lo anterior debe contemplarse al emplear programas que automáticamente asignen la cantidad y el tamaño de los paneles.

La técnica de combinar 2 aproximaciones con errores similares para obtener una aproximación más exacta puede aplicarse también a esta cuadratura. Por ejemplo, los valores de S(f) para dos cantidades diferentes de paneles, pueden combinarse para lograr un nuevo valor, el cual incluirá un error de h_i^7 y $f^{vi}(x)$. La sistematización de esta técnica desemboca en el popular método de la Cuadratura de Romberg.

V.5 CUADRATURA ADAPTADA.

Una Cuadratura Adaptada es un algoritmo numérico que utiliza una o dos cuadraturas básicas, determinando automáticamente el tamaño de los subintervalos o paneles, para aproximar la solución de una integral definida; cumpliéndose a la vez, con ciertos lineamientos de exactitud, previamente impuestos. El tamaño de cada panel depende de la *suavidad* de la curva a integrar. De esta forma, el tamaño del subintervalo será relativamente grande, donde la función tenga un comportamiento suave; y será pequeño, cuando ésta cambie abruptamente. Esto se aprecia en la Figura 5.6.

La mayor pérdida de tiempo de cómputo es debida al cálculo de f(x). Por lo tanto, si dos cuadraturas, brindan resultados que cumplen con la exactitud especificada, considerando la misma función en ambos casos, aquella que use el menor número de valuaciones de f(x), debe definirse como la más eficiente para ese problema en particular. Con la técnica descrita en el párrafo anterior, se logra un resultado que se apega a la exactitud deseada y que invierte el menor tiempo de cómputo posible.

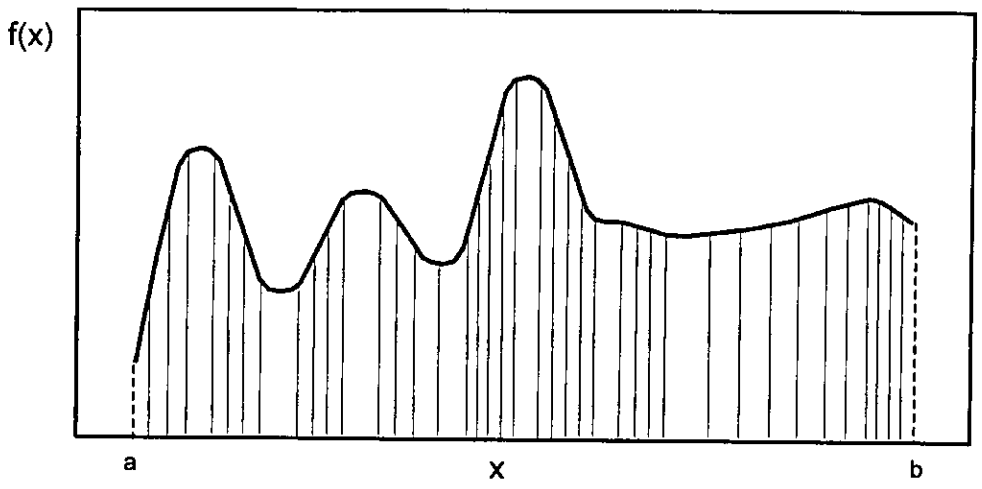


Figura 5.6. CUADRATURA ADAPTADA.

La Cuadratura Adaptada requiere de la información siguiente:

a) Un intervalo finito [a, b].
b) Un subprograma que valúe a f(x) para toda "x", x $\in$[a, b].
c) Una tolerancia de error (ε).

Entonces, el algoritmo calculará un valor Q, tal que:

$$\left| Q - \int_a^b f(x)dx \right| \leq \varepsilon \quad . \tag{5.47}$$

El intervalo de integración se divide en "n" paneles, $[x_i, x_{i+1}]$, de manera que, $x_i = a$ y $x_{i+1} = b$. Dicho número, está en función de la exactitud solicitada (ε) y de f(x). Como anteriormente fue definido, el ancho del panel es $h = x_{i+1} - x_i$.

Un esquema típico de Cuadratura Adaptada, aplica dos cuadraturas distintas en cada panel. Entonces, denotemos esos resultados como P_i y Q_i. Por ejemplo, un esquema que emplee la Regla de Simpson, se basa en la fórmula para dos paneles mostrada a continuación (ver Figura 5.7):

$$P_i = \frac{1}{6} h_i \left[f(x_i) + 4f\left(x_i + \frac{h_i}{2} \right) + f(x_i + h_i) \right] \quad , \tag{5.48}$$

y en la siguiente para cuatro paneles:

$$Q_i = \frac{h_i}{12} \left[f(x_i) + 4f\left(x_i + \frac{h_i}{4} \right) + 2f\left(x_i + \frac{h_i}{2} \right) + 4f\left(x_i + \frac{3h_i}{4} \right) + f(x_i + h_i) \right] \quad . \tag{5.49}$$

Tanto P_i como Q_i son aproximaciones a la integral:

$$I_i = \int_{x_i}^{x_{i+1}} f(x)\,dx \quad . \tag{5.50}$$

El principio fundamental de este esquema consiste en comparar las aproximaciones P_i y Q_i. Obteniéndose con esto, una estimación de la exactitud de tales aproximaciones, si el resultado es satisfactorio, una de ellas se considerará como el valor de la integral del subintervalo en análisis; de lo contrario, se dividirá el intervalo en dos o más partes y se repetirá el proceso en estos subintervalos más pequeños.

La evaluación de la función f(x) se simplifica debido a que tanto P_i como Q_i, poseen ciertos términos comunes entre sí. Q_i contiene únicamente dos términos extras que no están en P_i. De aquí en adelante, deberá suponerse que Q_i se obtiene aplicando la regla para P_i en dos ocasiones, una para cada mitad del intervalo. Esta es una técnica que facilitará el análisis de la Cuadratura Adaptada.

Considérese que la regla para P_i ofrece la respuesta exacta si la función por integrar es de grado p-1, o equivalentemente, si la p-ésima derivada, $f^{(p)}(x)$, es igual con cero.

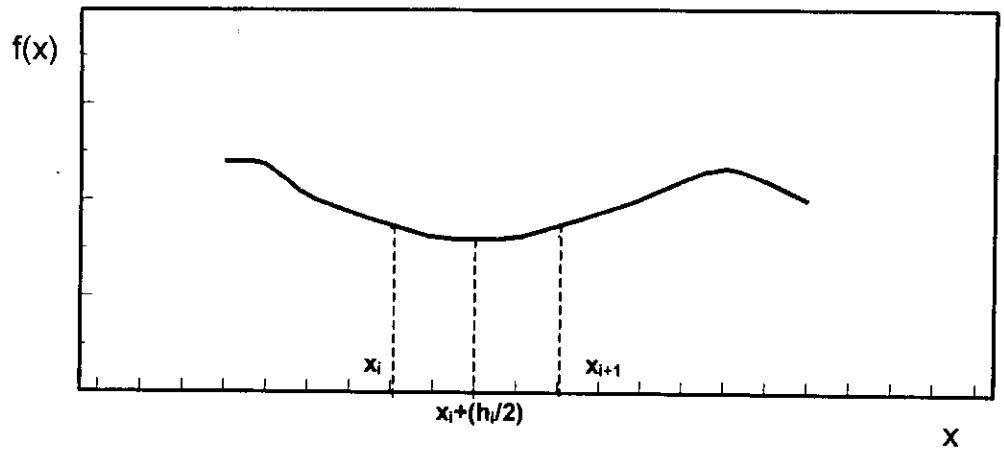


Figura 5.7. REGLA DE SIMPSON. CUADRATURA ADAPTADA.

Expandiendo la función por integrar en Series de Taylor, con respecto al punto medio del subintervalo, puede observarse que existe una constante "c", tal que:

$$I_i - P_i = c(h_i)^{p+1} f^{(p)}\left(x_i + \frac{h_i}{2}\right) + \ldots\ldots \quad . \tag{5.51}$$

Como se supuso que Q_i es la suma de dos P_i calculadas en subintervalos de longitud $h_i/2$, se tiene:

$$I_i - Q_i = c\left(\frac{h_i}{2}\right)^{p+1}\left[f^{(p)}\left(x_i + \frac{h_i}{4}\right) + f^{(p)}\left(x_i + \frac{3h_i}{4}\right)\right] + \ldots\ldots \quad , \tag{5.52}$$

pero como:

$$f^{(p)}\left(x_i + \frac{h_i}{4}\right) + f^{(p)}\left(x_i + \frac{3h_i}{4}\right) = 2f^{(p)}\left(x_i + \frac{h_i}{2}\right) + \ldots\ldots \quad , \tag{5.53}$$

entonces, los dos errores estarán relacionados por:

$$I_i - Q_i = \frac{2}{2^{p+1}}(I_i - P_i) + \ldots\ldots = \frac{1}{2^p}(I_i - P_i) + \ldots\ldots \quad . \tag{5.54}$$

Lo anterior indica que al bisectar el subintervalo, decrecerá el error en un factor de 2^p. Resolviendo para la incógnita I_i, y re-arreglando términos, se obtiene:

$$Q_i - I_i = \frac{1}{2^p - 1}(P_i - Q_i) + \ldots\ldots \quad . \tag{5.55}$$

En otras palabras, el error en la expresión Q_i es de alrededor de $1/(2^p - 1)$ veces la diferencia entre ambas aproximaciones.

El objetivo primario será entonces, bipartir cada subintervalo hasta que se satisfaga la siguiente desigualdad:

$$\frac{1}{2^p - 1}|P_i - Q_i| \le \frac{h_i}{b-a}\varepsilon \quad , \tag{5.56}$$

donde, ε es la tolerancia proporcionada por el usuario. Si el intervalo [a, b] fuese completamente cubierto por "n" subintervalos, el resultado sería:

$$Q = \sum_{i=1}^{n} Q_i \quad . \tag{5.57}$$

Considerando la igualdad anterior y despreciando los términos de alto orden, se tiene:

$$\begin{aligned} \left|Q - \int f(x)dx\right| &= \left|\sum_{i=1}^{n} Q_i - I_i\right| \\ &\le \sum_{i=1}^{n} |Q_i - I_1| \\ &\le \frac{1}{2^p - 1}\sum_{i=1}^{n} |P_i - Q_i| \\ &\le \frac{1}{2^p - 1}\left(\frac{2^p - 1}{b-a}\right)\varepsilon\sum_{i=1}^{n} h_i \\ &= \varepsilon \end{aligned} \quad , \tag{5.58}$$

lo cual coincide con lo planteado en la ecuación (5.47).

Debe recordarse que este análisis requiere la suposición de que $f^{(p)}(x)$ es continua, y que el error es proporcional a $(h_i)^{p+1} f^{(p)}(x)$. Aunque en la realidad esto no se cumpla, la cuadratura arrojará finalmente, un resultado apegado a la tolerancia fijada.

Por simplicidad, se ha manejado el criterio de error absoluto, o sea:

$$\left|Q - \int f\right| \leq \varepsilon \quad . \tag{5.59}$$

En numerosas ocasiones, se utiliza algún tipo de error relativo, el cual es independiente de factores de escala en la función f, complicándose su verificación por diversas razones. Un criterio de error relativo puro, es:

$$\frac{\left|Q - \int f\right|}{\left|\int f\right|} \leq \varepsilon \quad . \tag{5.60}$$

Como el denominador, $\int f$, puede ser cero, debido a la cancelación asociada a integrales oscilantes, el criterio puede ser imposible de satisfacer. Además, una buena aproximación del valor del denominador se lograría sólo hasta el final del cálculo.

Otras cuadraturas, usan un criterio de error relativo que involucra a $\int |f|$, es decir:

$$\frac{\left|Q - \int f\right|}{\int |f|} \leq \varepsilon \quad . \tag{5.61}$$

En la expresión anterior, el denominador será distinto de cero, a menos que, la función f fuese idéntica a cero.

El usuario de una subrutina de Cuadratura Adaptada debe tener en mente estas pruebas de exactitud, y por supuesto, debe revisar la documentación particular para tal subrutina.

V.5.1 SUBRUTINA QUANC8.

QUANC8 es una subrutina que emplea una técnica de cuadratura adaptada para aproximar el valor de una integral definida. Su nombre se deriva de Cuadratura Adaptada de Newton-Cotes para 8 paneles (*Quadrature Adaptive of Newton-Cotes 8-panel*). Las fórmulas de Newton-Cotes constituyen una familia de cuadraturas o reglas para integración e interpolación polinomial, sobre puntos de evaluación igualmente espaciados. Las reglas del Rectángulo, del Trapezoide y de Simpson, son obtenidas integrando polinomios de 0, 1° y 2° grado; y son los primeros miembros de la familia. La fórmula de 8 paneles requiere la integración de un polinomio de 8° grado, pero igual que para la regla de Rectángulo o de Simpson, se permite un grado *extra*, de manera que, para polinomios de 9° grado se tendrán resultados exactos.

La subrutina se presenta, en lenguaje *Visual Basic*, en el CD de consulta (Quanc8.exe), e incluye un subprograma, FUN(X), para valuar la integral f(x); el usuario debe proporcionar los límites, superior e inferior, de integración, A = a y B = b, y las tolerancias de error absoluto, ABSERR, y relativo, RELERR. La subrutina calcula la aproximación de la integral, RESULT, el error absoluto estimado, ERREST, el número de valuaciones requeridas de f(x), NOFUN, y un indicador de la eficacia del cálculo, FLAG. Si FLAG es cero, RESULT cumplirá con la exactitud fijada; si FLAG es un valor mínimo, RESULT será aceptable, y en caso contrario, será inaceptable.

Como esta regla es exacta para polinomios de 9° grado, el análisis de la sección V.5 puede aplicarse con p = 10. El error en Q_i es del orden de $1/(2^{10} - 1) = 1/1023$ veces el error en P_i, sólo si la curva por integrar es suave y se desprecian los términos de alto grado.

Entonces, los resultados de un subintervalo en particular son aceptables, si:

$$\frac{1}{1023}|P_i - Q_i| \le \frac{h_i}{b-a}\varepsilon \quad . \tag{5.62}$$

Cada subintervalo será bisectado, existiendo una cantidad límite, LEVMAX, la cual es de 30 biparticiones. Cuando tal límite es sobrepasado, el subintervalo se aprobará siempre y cuando se cumpla el criterio de exactitud establecido, pero dicho número se almacenará en la parte entera de FLAG. Por lo tanto, la parte entera del valor de FLAG, representa el número de subintervalos o paneles que se calcularon al superarse LEVMAX, y la parte decimal, indica el porcentaje de análisis del intervalo en el que se detecte la no convergencia a la tolerancia. También, puede averiguarse en que valor de la abscisa, x, se presentó el problema, mediante la siguiente relación:

$$X0 = B - (\text{parte fraccional de FLAG}) \times (B - A)$$

Por ejemplo, si A = 0, B = 2 y FLAG = 91.21, 91 subintervalos fueron obtenidos, lográndose analizar el 21% del intervalo que muestra dificultades. Y el valor de "x" para el cual se detectó la no convergencia es X0 = 2 - 0.21 × (2-1) = 1.58.

V.6 CUADRATURAS GAUSSIANAS.

Otra alternativa para obtener la aproximación de una integral definida la representan las Cuadraturas Gaussianas, que involucran una sumatoria ponderada de la función, evaluada en los puntos base, esto es:

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n} w_i f(x_i) \quad , \tag{5.63}$$

donde w_i es el peso o ponderador asociado a cada valor de la función $f(x_i)$, i = 0, 1,...,n. Antes de proceder al desarrollo de tales cuadraturas es necesario exponer el tema de los Polinomios Ortogonales.

V.6.1 POLINOMIOS ORTOGONALES.

Dos funciones, $g_n(x)$ y $g_m(x)$, seleccionadas de una familia de funciones $\{g_k(x)\}$, son ortogonales, con respecto a la función ponderadora, w(x), en el intervalo [a, b], si:

$$\int_a^b w(x)\, g_n(x) g_m(x) dx = 0 \quad , \quad n \neq m \quad , \tag{5.64}$$

$$\int_a^b w(x) \left[g_n(x)\right]^2 dx = c(n) \neq o \quad . \tag{5.65}$$

En general, "c" depende de "n". Si estas relaciones se cumplen para toda "n", la familia de funciones, $\{g_k(x)\}$, constituye un conjunto de funciones ortogonales. Las funciones ortogonales más comunes son {sen kx} y {cos kx}.

La ortogonalidad, en términos de espacios vectoriales, se interpreta como la perpendicularidad existente entre dos vectores, en un espacio de "n" dimensiones, siendo "n" un valor muy grande (ver la Figura 5.8).

La familia de polinomios $\{x^k\}$, define un espacio vectorial cuyos elementos no son ortogonales. Otras familias, como las constituidas por polinomios de Legendre, de Laguerre, de Chebyshev o de Hermite, definen un espacio vectorial y además son ortogonales.

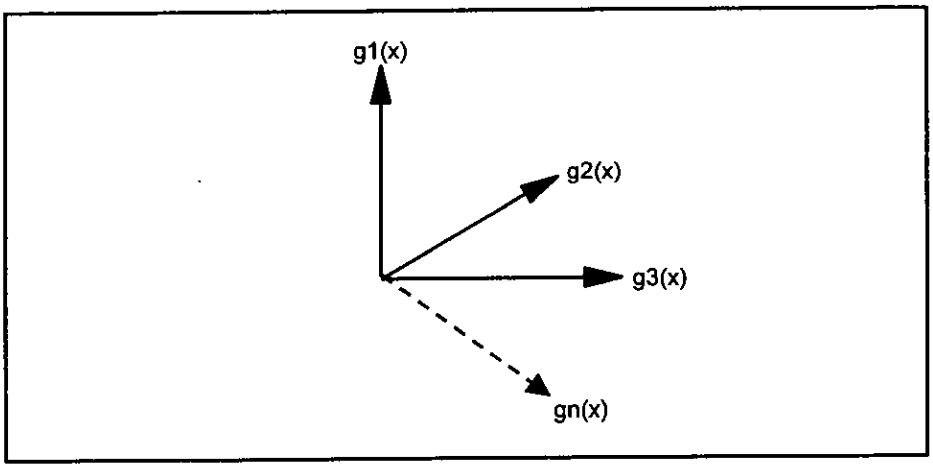


Figura 5.8. FAMILIA DE FUNCIONES ORTOGONALES.

CADA FUNCIÓN ESTÁ REPRESENTADA POR UN VECTOR EN UN ESPACIO n-DIMENSIONAL.

Enseguida, se enuncian los principios básicos de las familias de polinomios ortogonales mencionadas:

a) Polinomios de Legendre {$P_n(x)$}.

Los polinomios de Legendre son ortogonales en el intervalo [-1, 1], y su función ponderadora es w(x) = 1, esto es:

$$\int_{-1}^{1} P_n(x)P_m(x)dx = 0 \quad , \quad n \neq m \quad , \tag{5.66}$$

$$\int_{-1}^{1} [P_n(x)]^2 \, dx = c(n) \neq 0 \quad . \tag{5.67}$$

La ecuación de recurrencia es:

$$P_n(x) = \frac{(2n-1)x}{n} P_{n-1}(x) - \frac{n-1}{n} P_{n-2}(x) \quad , \tag{5.68}$$

los primeros polinomios de Legendre son:

$$P_0(x) = 1 \quad , \tag{5.69}$$

$$P_1(x) = x \quad , \tag{5.70}$$

$$P_2(x) = \frac{1}{2}(3x^2 - 1) \quad . \tag{5.71}$$

b) Polinomios de Laguerre {$L_n(x)$}.

Los polinomios de Laguerre son ortogonales en el intervalo [0, ∞], y su función ponderadora es $w(x) = e^{-x}$, por lo que:

$$\int_0^{\infty} e^{-x} L_n(x) L_m(x)\,dx = 0 \quad , \quad n \neq m \quad , \tag{5.72}$$

$$\int_0^{\infty} e^{-x} [L_n(x)]^2\,dx = c(n) \neq 0 \quad . \tag{5.73}$$

La ecuación de recurrencia es:

$$L_n(x) = (2n - x - 1) L_{n-1}(x) - (n^2 - 1) L_{n-2}(x) \quad , \tag{5.74}$$

los primeros polinomios de Laguerre son:

$$L_0(x) = 1 \quad , \tag{5.75}$$

$$L_1(x) = -x + 1 \quad , \tag{5.76}$$

$$L_2(x) = x^2 - 4x + 2 \quad . \tag{5.77}$$

c) Polinomios de Chebyshev {$T_n(x)$}.

Los polinomios de Chebyshev son ortogonales en el intervalo [-1, 1], y su función ponderadora es $w(x) = 1/\sqrt{1 - x^2}$, o sea:

$$\int_{-1}^{1} \left[1/\sqrt{1 - x^2} \right] T_n(x) T_m(x)\,dx = 0 \quad , \quad n \neq m \quad , \tag{5.78}$$

$$\int_{-1}^{1} \left[1/\sqrt{1 - x^2} \right] [T_n(x)]^2\,dx = c(n) \neq 0 \quad . \tag{5.79}$$

La ecuación de recurrencia es:

$$T_n(x) = 2x\, T_{n-1}(x) - T_{n-2}(x) \quad , \tag{5.80}$$

los primeros polinomios de Legendre son:

$$T_0(x)=1 \quad , \tag{5.81}$$
$$T_1(x)=x \quad , \tag{5.82}$$
$$T_2(x)=2x^2-1 \quad . \tag{5.83}$$

d) Polinomios de Hermite {$H_n(x)$}.

Los polinomios de Hermite son ortogonales en el intervalo [-∞, ∞], y su función ponderadora es $w(x)=e^{-x^2}$, es decir:

$$\int_{-\infty}^{\infty} e^{-x^2} H_n(x) H_m(x)\,dx \quad , \quad n \neq m \quad , \tag{5.84}$$
$$\int_{-\infty}^{\infty} e^{-x^2} [H_n(x)]^2\,dx = c(n) \neq 0 \quad . \tag{5.85}$$

La ecuación de recurrencia es:

$$H_n(x)=2x\,H_{n-1}(x)-2(n-1)H_{n-2}(x) \quad , \tag{5.86}$$

los primeros polinomios de Hermite son:

$$H_0(x)=1 \quad , \tag{5.87}$$
$$H_1(x)=2x \quad , \tag{5.88}$$
$$H_2(x)=4x^2-2 \quad . \tag{5.89}$$

Cada uno de los polinomios $P_n(x)$, $L_n(x)$, $T_n(x)$ y $H_n(x)$, es un polinomio con coeficientes reales, de n-ésimo grado en la variable "x", y con "n" raíces reales distintas dentro del intervalo de integración apropiado. Por ejemplo, todas las raíces de $L_n(x)$ están dentro del intervalo [0, ∞].

Un polinomio cualquiera, de grado n-ésimo, $p_n(x)=\sum_{i=0}^{n}\alpha_i x^i$, puede ser representado como una combinación lineal de cualquier familia de polinomios ortogonales. Por lo que:

$$p_n(x)=\beta_0 Z_0(x)+\beta_1 Z_1(x)+.....+\beta_n Z_n(x)=\sum_{i=1}^{n}\beta_i Z_i(x) \quad , \tag{5.90}$$

donde $Z_i(x)$ es un polinomio de i-ésimo grado de alguna familia de polinomios ortogonales, obtenido con la ecuación de recurrencia respectiva.

Por ejemplo, si expandemos un polinomio de cuarto grado, $p_4(x)$, en términos de los polinomios de Legendre:

$$p_4(x)=\sum_{i=0}^{n}\alpha_i x^i=\sum_{i=o}^{n}\beta_i Z_i=\beta_0+\beta_1 x+\beta_2\left[\frac{3}{2}x^2-\frac{1}{2}\right]+\beta_3\left[\frac{5}{2}x^3-\frac{3}{2}x\right]+\beta_4\left[\frac{35}{8}x^4-\frac{15}{4}x^2+\frac{3}{8}\right] \quad , \tag{5.91}$$

agrupando términos:

$$p_4(x)=\left[\beta_0-\frac{1}{2}\beta_2 x+\frac{3}{8}\beta_4\right]+\left[\beta_1-\frac{3}{2}\beta_3\right]x+\left[\frac{3}{2}\beta_2-\frac{15}{4}\beta_4\right]x^2+\frac{5}{2}\beta_3 x^3+\frac{35}{8}\beta_4 x^4 \quad , \tag{5.92}$$

igualando con los coeficientes α_i, se tiene:

$$\beta_4=\frac{8}{35}\alpha_4 \quad , \tag{5.93}$$

$$\beta_3=\frac{2}{5}\alpha_3 \quad , \tag{5.94}$$

$$\beta_2=\frac{2}{3}\left[\alpha_2+\frac{6}{7}\alpha_4\right] \quad , \tag{5.95}$$

$$\beta_1=\alpha_1+\frac{3}{5}\alpha_3 \quad , \tag{5.96}$$

$$\beta_0=\alpha_0+\frac{1}{3}\alpha_2+\frac{1}{5}\alpha_4 \quad . \tag{5.97}$$

El polinomio $p_4(x) = x^4 + 3x^3 - 2x^2 + 2x$, expresado por polinomios de Legendre (el lector puede verificarlo), es:

$$p_4(x) = -\frac{22}{15}P_0(x) + \frac{19}{5}P_1(x) - \frac{16}{21}P_2(x) + \frac{6}{5}P_3(x) + \frac{8}{35}P_4(x) \quad . \tag{5.98}$$

V.6.2 CUADRATURA DE GAUSS-LEGENDRE.

El método de las Cuadraturas Gaussianas propone aproximar una integral definida, por medio de la siguiente relación matemática:

$$\int_a^b f(x)\,dx = \int_a^b w(x)f(x)\,dx \approx \sum_{i=0}^{n} w_i f(t_i) \quad . \tag{5.99}$$

Siendo w(x) la función peso asociada a alguna familia de polinomios ortogonales, cada nodo t_i es la raíz de un polinomio de grado "n" de tal familia, y cada ponderador w_i es la solución de la integral, efectuada mediante un polinomio de grado "i", de la misma familia, sobre el intervalo [a, b].

Muchas técnicas de integración numérica, como las cuadraturas del Rectángulo, del Trapecio, de Simpson y Spline, emplean puntos o nodos equidistantes, x_i, lo cual puede ser conveniente pero no necesario, además w(x) = 1. En contraste, en los métodos Gaussianos no se requiere de puntos igualmente espaciados, sino que se usan las raíces de un polinomio ortogonal de grado "n", definido en el intervalo [a, b], y con la función w(x) como función peso.

El método de Gauss-Legendre se apoya en los polinomios de Legendre para dar solución a una integral definida; la ecuación respectiva es:

$$\int_{-1}^{1} F(z)\,dz \approx \sum_{i=o}^{n} w_i F(z_i) \quad , \tag{5.100}$$

donde:

$$w_i = \int_{-1}^{1} L_i(z)\,dz = \int_{-1}^{1} \prod_{\substack{j=c \\ j\neq i}}^{n} \left[\frac{z - z_j}{z_i - z_j}\right] dz \quad . \tag{5.101}$$

Siendo z_i la i-ésima raíz de un polinomio de Legendre de grado n+1, $P_{n+1}(z)$. Dichas raíces pueden calcularse mediante la siguiente expresión:

$$\prod_{i=o}^{n} (z - z_i) = b_{n+1} P_{n+1}(z) \quad . \tag{5.102}$$

Algunos valores de z_i y de w_i para n = 1, 2, 3 y 4 (correspondientes a las fórmulas para 2, 3, 4 y 5 puntos, respectivamente) se presentan en la Tabla 5.1[(2)].

Raíces (z_i)	$\int_{-1}^{1} F(z)\,dz \approx \sum_{i=0}^{n} w_i F(z_i)$	Factores (w_i)
± 0.5773502692	Fórmula de dos puntos n = 1	1.0000000000
0.0000000000 ± 0.7745966692	Fórmula de tres puntos n = 2	0.8888888889 0.5555555556
± 0.3399810436 ± 0.8611363116	Fórmula de cuatro puntos n = 3	0.6521451549 0.3478548451
0.0000000000 ± 0.5384693101 ± 0.9061798459	Fórmula de cinco puntos n = 4	0.5688888889 0.4786286704 0.2369268850

Tabla 5.1. RAÍCES DE POLINOMIOS DE LEGENDRE, $P_{n+1}(x)$ y, FACTORES DE PESO PARA LA CUADRATURA DE GAUSS-LEGENDRE.

Como puede observarse, el método de Gauss-Lagendre es aplicable en el intervalo $-1 \leq z \leq 1$, para extenderlo al intervalo $a \leq x \leq b$, se puede implementar un cambio de variable, es decir:

$$x = \frac{z(b-a) + a + b}{2} \quad , \tag{5.103}$$

de tal manera que:

$$\int_a^b f(x)dx = \frac{b-a}{2}\int_{-1}^{1} f\left[\frac{z(b-a)+a+b}{2}\right]dz \quad , \tag{5.104}$$

entonces, considerando la expresión (5.100), se llega a:

$$\int_a^b f(x)dx = \frac{b-a}{2}\sum_{i=0}^{n} w_i f\left[\frac{z_i(b-a)+a+b}{2}\right] \quad . \tag{5.105}$$

La ecuación (5.105) es la fórmula general de la Cuadratura de Gauss-Legendre, y es más apropiada que la expresión (5.100) para cálculos mediante computadoras, puesto que no requiere de una transformación simbólica de f(x); en lugar de ella, los puntos base, z_i, son modificados y los factores de peso, w_i, son afectados por (b-a)/2.

Las expresiones para esta cuadratura son exactas si la función a integrar es un polinomio de grado menor o igual a 2n+1, en caso contrario, el error en que se incurre está dado por:

$$E_n(z) = \frac{2^{2n+3}\left[(n+1)!\right]^4}{(2n+3)\left[(2n+2)!\right]^3} F^{2n+2}(\xi) \quad , \quad \xi \in (-1,\ 1) \quad . \tag{5.106}$$

Si las magnitudes de las derivadas de alto orden de F(z), disminuyen o no se incrementan substancialmente, mientras que "n" crece, la Cuadratura de Gauss-Legendre será más precisa que otras Cuadraturas Gaussianas (que se describirán a continuación), para valores equivalentes de "n".

V.6.3 CUADRATURA DE GAUSS-LAGUERRE.

Los polinomios de Laguerre son la herramienta de la Cuadratura de Gauss-Laguerre. La ecuación de esta cuadratura es:

$$\int_0^{\infty} e^{-z} F(z)\,dz \approx \sum w_i F(z_i) \quad , \tag{5.107}$$

donde:

$$w_i = \int_0^{\infty} e^{-z} L_i(z)\,dz = \int_0^{\infty} e^{-z} \prod_{\substack{j=0 \\ j\neq i}}^{n} \left[\frac{z - z_j}{z_i - z_j} \right] dz \quad . \tag{5.108}$$

Siendo z_i la i-ésima raíz de un polinomio de Laguerre de grado n+1, $L_{n+1}(z)$. Dichas raíces pueden calcularse mediante la expresión:

$$\prod_{i=0}^{n} (z - z_i) = b_{n+1} L_{n+1}(z) \quad . \tag{5.109}$$

Los valores de z_i y de w_i para n = 1, 2, 3 y 4 (correspondientes a las fórmulas para 2, 3, 4 y 5 puntos, respectivamente) se presentan en la Tabla 5.2[(2)].

Las ecuaciones para esta cuadratura son exactas si la función a integrar es un polinomio de grado menor o igual a 2n+1, en caso contrario, el error en que se incurre está dado por:

$$E_n = \frac{[(n+1)!]^2}{(2n+2)!} F^{2n+2}(\xi) \quad , \quad \xi \in (0, \infty) \quad . \tag{5.110}$$

Raíces (z_i)	$\int_0^{\infty} e^{-z}F(z)dz \approx \sum_{i=0}^{n} w_i F(z_i)$	Factores (w_i)
0.5857864376 3.4142135624	Fórmula de dos puntos n = 1	0.8535533906 0.1464466094
0.4157745568 2.2942803603 6.2899450829	Fórmula de tres puntos n = 2	0.7110930099 0.2785177336 0.0103892565
0.3225476896 1.7457611012 4.5366202969 9.3950709123	Fórmula de cuatro puntos n = 3	0.6031541043 0.3574186924 0.0388879085 0.0005392947
0.2635603197 1.4134030591 3.5964257710 7.0858100059 12.6408008442	Fórmula de cinco puntos n = 4	0.5217556106 0.3986668111 0.0759424497 0.0036117587 0.0000233700

Tabla 5.2. RAÍCES DE POLINOMIOS DE LAGUERRE, $L_{n+1}(z)$ y, FACTORES DE PESO PARA LA CUADRATURA DE GAUSS-LAGUERRE.

Para evaluar integrales de la forma:

$$\int_a^{\infty} e^{-x}f(x)dx \quad , \tag{5.111}$$

donde el valor del límite inferior "a", es arbitrario y finito, debe realizarse un cambio lineal de variable, x = z + a. Por lo tanto:

$$\int_a^{\infty} e^{-x}f(x)dx = \int_0^{\infty} e^{-(z+a)}f(z+a)dz = e^{-a}\int_0^{\infty} e^{-z}f(z+a)dz \quad , \tag{5.112}$$

entonces, la expresión (5.107) se transformará en:

$$\int_a^{\infty} e^{-x}f(x)dx \approx e^{-a}\sum_{i=0}^{n} w_i f(z_i + a) \quad . \tag{5.113}$$

V.6.4 CUADRATURA DE GAUSS-CHEBYSHEV.

La Cuadratura de Gauss-Chebyshev se basa en los polinomios ortogonales de Chebyshev. La ecuación para esta cuadratura es:

$$\int_{-1}^{1}\left[1/\sqrt{1-z^2}\right]F(z)\,dz \approx \sum_{i=o}^{n} w_i F(z_i) \quad , \tag{5.114}$$

donde los n+1 valores z_i, son las raíces de polinomios de Chebyshev de grado n+1, $T_{n+1}(z)$. Por lo cual:

$$z_i = \cos\left[\frac{(2i+1)\pi}{(2n+2)}\right] \quad , \quad i = 0,1,....,n \quad . \tag{5.115}$$

En este caso, las w_i son todas iguales y tienen el valor $\pi/(n+1)$. Entonces la ecuación (5.114) se simplifica, quedando de la manera siguiente:

$$\int_{-1}^{1}\left[1/\sqrt{1-z^2}\right]F(z)\,dz \approx \frac{\pi}{n+1}\sum_{i=o}^{n} F(z_i) \quad . \tag{5.116}$$

Las expresiones para esta cuadratura son exactas si la función a integrar es un polinomio de grado menor o igual a 2n+1, en caso contrario, el error en que se incurre está dado por:

$$E_n = \frac{2\pi}{2^{2n+2}(2n+2)!}F^{2n+2}(\xi) \quad , \quad \xi \in (-1,1) \quad . \tag{5.117}$$

Es evidente que el método de Gauss-Chebyshev es aplicable en el intervalo $-1 \le z \le 1$, para extenderlo al intervalo $a \le x \le b$, debe realizarse un cambio de variable:

$$x = \frac{z(b-a)+a+b}{2} \quad , \tag{5.118}$$

de tal manera que:

$$\int_a^b f(x)dx = \frac{b-a}{2}\int_{-1}^{1} f\left[\frac{z(b-a)+a+b}{2}\right]dz \quad . \tag{5.119}$$

Rescribiendo la expresión anterior y considerando la función ponderadora de los polinomios de Chebyshev, $w(x) = 1/\sqrt{1-x^2}$, se llega a:

$$\int_a^b f(x)dx = \frac{b-a}{2}\int_{-1}^{1}\left[\frac{1}{\sqrt{1-z^2}}\right]F(z)dz \quad , \tag{5.120}$$

donde:

$$F(z) = \sqrt{1-z^2}\, f\left[\frac{z(b-a)+a+b}{2}\right] \quad . \tag{5.121}$$

La ecuación (5.116) puede transformarse en:

$$\int_a^b f(x)dx \approx \frac{\pi(b-a)}{2m}\sum_{i=1}^{m}\sqrt{1-z_i^2}\, f(x_i) \quad , \tag{5.122}$$

donde:

$$z_i = \cos\left(\frac{[2(i-1)+1]\pi}{2m}\right) \quad , \tag{5.123}$$

$$i = 1,2,.....,m \quad ,$$

$$x_i = \frac{z_i(b-a)+a+b}{2} \quad . \tag{5.124}$$

Aquí, "m" es el número de puntos a usar en la cuadratura, y x_i es simplemente una raíz z_i, de un polinomio de Chebyshev de m-ésimo grado transformado para el intervalo [a, b].

V.6.5 CUADRATURA DE GAUSS-HERMITE.

La Cuadratura de Gauss-Hermite se basa en los polinomios ortogonales de Hermite. La expresión de esta cuadratura es:

$$\int_{-\infty}^{\infty} e^{-x^2} f(x)\,dx \approx \sum_{i=0}^{n} w_i f(x_i) \quad . \tag{5.125}$$

Las x_i son las raíces de un polinomio de Hermite de grado n+1. Los valores de z_i y de w_i para n = 1, 2, 3 y 4 (correspondientes a las fórmulas para 2, 3, 4 y 5 puntos, respectivamente) se presentan en la Tabla 5.3[2].

Raíces (z_i)	$\int_{-\infty}^{\infty} e^{-x^2} f(x)\,dx \approx \sum_{i=0}^{n} w_i f(x_i)$	Factores (w_i)
± 0.7071067811	Fórmula de dos puntos n = 1	0.8862269255
± 1.2247448714 0.0000000000	Fórmula de tres puntos n = 2	0.2954089752 1.1816359006
± 1.6506801239 ± 0.5246476233	Fórmula de cuatro puntos n = 3	0.0813128354 0.8049140900
± 1.0201828705 ± 0.9585725646 0.0000000000	Fórmula de cinco puntos n = 4	0.0199532421 0.3936193232 0.9453087205

Tabla 5.3. RAÍCES DE POLINOMIOS DE HERMITE, $H_{n+1}(x)$ y, FACTORES DE PESO PARA LA CUADRATURA DE GAUSS-HERMITE.

Como en los casos anteriores, las expresiones para esta cuadratura son exactas si la función a integrar es un polinomio de grado menor o igual a 2n+1, en caso contrario, el error en que se incurre está dado por:

$$E_n = \frac{(n+1)!\sqrt{\pi}}{2^{2n+2}(2n+2)!} F^{2n+2}(\xi) \quad , \quad \xi \in (-\infty, \infty) \quad . \tag{5.126}$$

La subrutina GAUSSQ (en lenguaje *Visual Basic*), se incluye en el CD de consulta (Gaussq.exe), y permite elegir alguna de las Cuadraturas Gaussianas expuestas, para dar solución a una integral definida.

V.7 CÁLCULO DEL POTENCIAL O PSEUDO PRESIÓN DE LOS GASES REALES.

En 1966 Al-Hussainy y colaboradores[14], demostraron que una integral (que en lo sucesivo denominaremos *Potencial* o *Pseudo Presión de los Gases Reales*), puede usarse para cálculos ingenieriles que incorporen cambios en la viscosidad y compresibilidad de un gas. El potencial de gas real está definido como:

$$m(p)=2\int_{p_m}^{p}\frac{p}{\mu(p)Z(p)}dp \quad , \tag{5.127}$$

donde p_m es la presión base. El potencial de gas real tiene las unidades de $[(lb/pg^2\ abs)^2/cp]$. Para flujo isotérmico de gas, la viscosidad, μ, y el factor de compresibilidad, Z, son función de la presión, por lo que serán los únicos parámetros involucrados en la obtención del m(p). Entonces, empleando correlaciones de propiedades físicas del gas, la función m(p) se solucionará fácilmente.

Al-Hussainy y colaboradores, también indicaron que el potencial de gas real puede usarse en el análisis de pruebas de presión en pozos de gas; coadyuvando al cálculo de la permeabilidad y del factor de daño, por citar algunos conceptos.

El programa propuesto facilita la solución del potencial de gas real, a cualquier presión y temperatura, además, permite almacenar la información generada, en archivos para su lectura o escritura continua (archivos con extensión dat). El lenguaje de programación adoptado es el *Visual Basic.* Las Cuadraturas implementadas en el programa de cómputo son las siguientes: la del Rectángulo, la del Trapecio, la de Simpson, la Adaptada (Quanc8) y Spline. El programa se presenta en el CD de consulta (m(p).exe, consultar Apéndice A), y el diagrama de bloques en la página 180.

La ejecución del programa requiere de datos de temperatura del yacimiento, gravedad específica del gas y presión a la que se desea obtener el m(p). Los resultados se despliegan de manera tabular, y se considerará como la solución más exacta la ofrecida por QUANC8 (debido a las "bondades" que posee, consultar sección V.5.1), entonces, el error relativo se calculará en base al resultado arrojado por dicha subrutina.

Es oportuno señalar que la subrutina QUANC8 incluye una modificación, la cual consiste en el reemplazo de la función FUN por la función SEVAL (y por ende se usará la subrutina SPLINE), para evaluar, mediante una interpolación, la integral en el punto deseado; incluso en aquellas técnicas que requieren evaluar la integral en presiones intermedias, se emplea la función SEVAL para tal fin (consultar listado del programa). El factor de compresibilidad del gas, Z, se calcula mediante la correlación de Hall-Yarborough[14], la cual contempla gases con contenidos mínimos de impurezas (N_2, CO_2 y/o H_2S); y su viscosidad, se obtiene con la correlación de Lee y colaboradores[12].

DATOS A CONSIDERAR.

La información usada en este ejemplo es[14]: temperatura del yacimiento de 200 °F y gravedad específica del gas igual a 0.85 (el gas está libre de impurezas).

Obtenga el potencial de gas real, m(p), a la presión de 4400 [lb/pg^2 abs].

RESULTADOS.

El resultado publicado en el trabajo original[14], que implica el uso de correlaciones gráficas, es el siguiente:

$$m(4,400) = 1{,}107.285 \times 10^6 \left[\frac{(lb/pg^2 abs)^2}{cp} \right] .$$

Los resultados logrados con el programa de cómputo se exponen en la Tabla 5.4. En ella puede apreciarse gran concordancia entre los valores obtenidos por las cuadraturas. El máximo error relativo, con respecto al valor del publicado original, lo presenta el método del Trapecio, debido a la metodología propia de la técnica (consultar sección V.2); la magnitud de tal error es:

$$\text{Error Relativo Máximo} = \left|\frac{1{,}060.002 - 1{,}107.285}{1{,}107.285}\right| \times 100 = 4.27\% \quad .$$

Se puede concluir que los resultados ofrecidos por el programa son buenos, además de ser obtenidos de manera rápida y descartando el uso de correlaciones gráficas (que en muchos casos tienen una deficiente impresión). En la Figura 5.10 se muestra una gráfica de p vs. $p/(\mu\ Z)$, los datos se tomaron de las bases de datos generadas durante la ejecución del programa. El área bajo la curva representa el valor de m(p) en el intervalo de presión de 0 a 4400 [lb/pg^2 abs].

Técnica	m(4,400 psia) [(lb/pg² abs)²/cp]	Error Relativo con respecto a QUANC8 [%]
QUANC8	1.060272×10^{9}	----------
Rectángulo	1.060474×10^{9}	1.903992×10^{-02}
De Simpson	1.060317×10^{9}	4.208825×10^{-03}
SPLINE	1.060317×10^{9}	4.208825×10^{-03}
Del Trapecio	1.060002×10^{9}	2.544158×10^{-02}

Tabla 5.4. RESULTADOS DEL CÁLCULO DEL m(p).

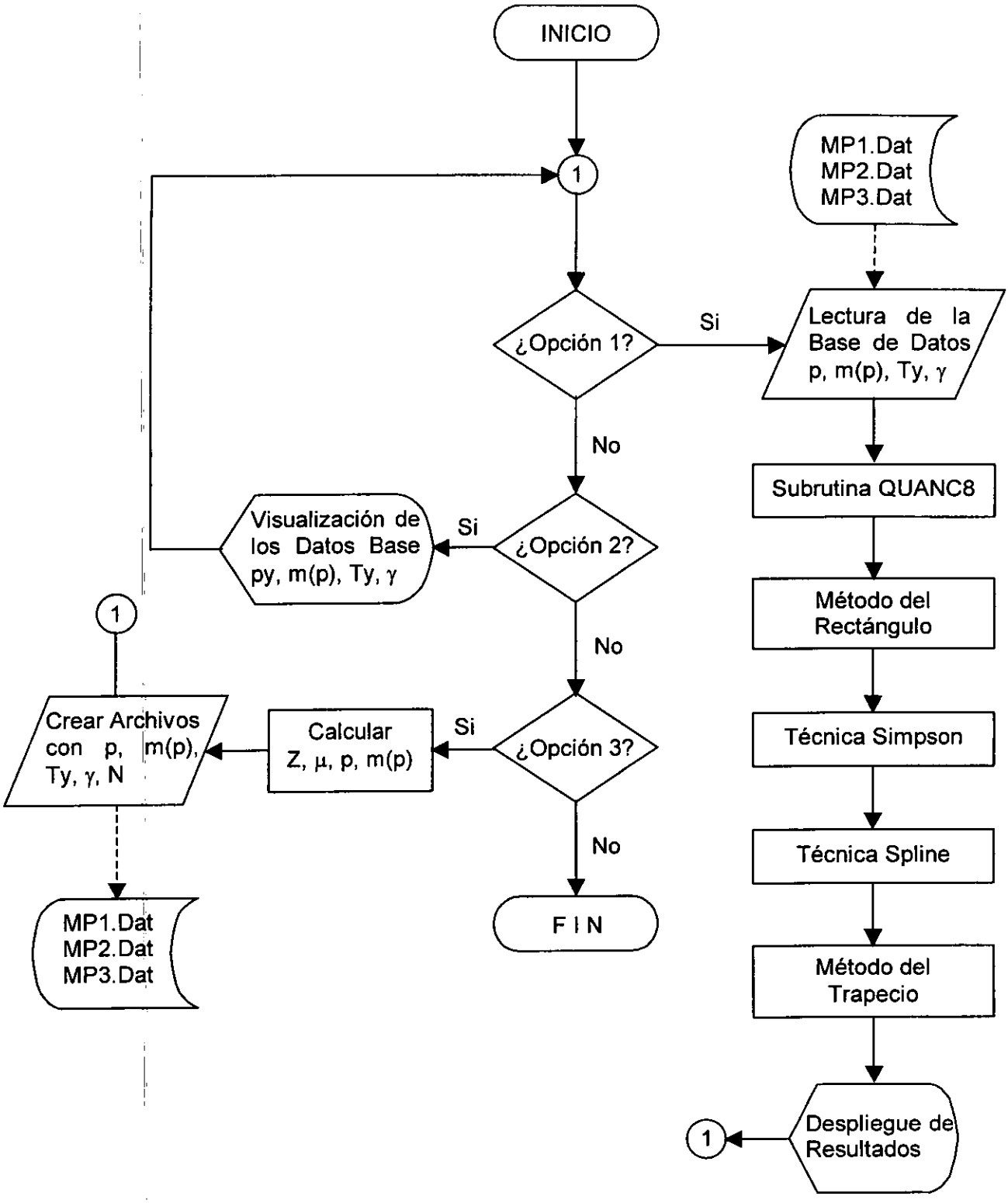


Figura 5.9. DIAGRAMA DE BLOQUES PARA OBTENER EL POTENCIAL DE GAS REAL, m(p).

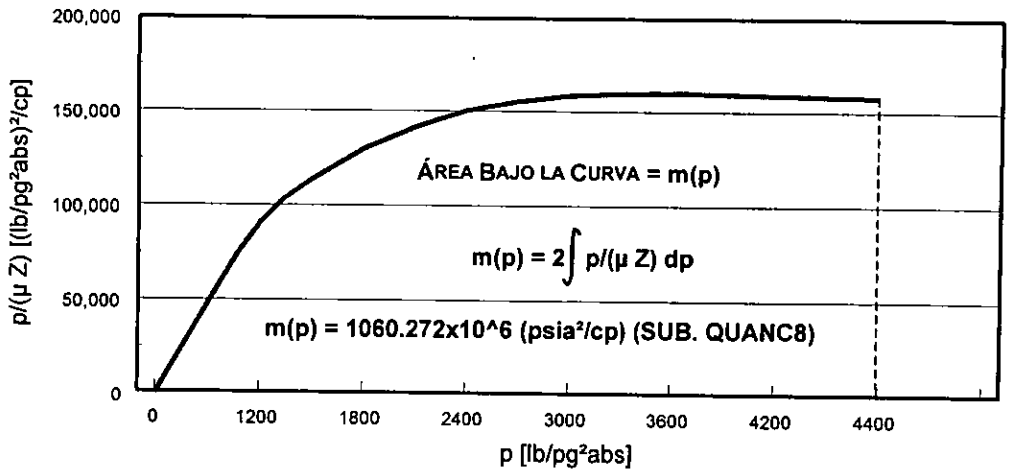


Figura 5.10. RESULTADOS DEL CÁLCULO DEL m(p), T=200°F Y SG=0.85.

V.8 PROBLEMAS PROPUESTOS.

Problema Propuesto V.1. Gill y Scher*, modificando la fórmula de Prandtl, encontraron una expresión para la distribución de la velocidad de un fluido en una tubería de sección circular. La velocidad adimensional u^+, en la dirección x, está dada como una función de la distancia adimensional, y^+, medida a partir de la pared interna de la tubería (ver Figura 5.11), por medio de la siguiente ecuación:

$$u^+ = \int_0^{y^+} \frac{-1+\sqrt{1+4cd}}{2c}\,dy^+ = \int_0^{y^+} \frac{2d}{1+\sqrt{1+4cd}}\,dy^+ \quad ,$$

donde:

$$c = 0.129\left(y^+\right)^2\left(1-e^{-\theta\left(y^+/\bar{y}^+\right)}\right)^2 \quad ,$$

y:

$$d = 1-\frac{y^+}{\bar{y}^+} \quad .$$

* W. N. GILL Y M. SCHER, "A MODIFICATION OF THE MOMENTUM TRANSPORT HYPOTHESIS", A. I. CH. E. JOURNAL, 7, PAGS. 61-65, 1961

Aquí, $\bar{y}^+$ es la distancia adimensional correspondiente al centro de la tubería (eje de la tubería), y viene dada por:

$$\bar{y}^+ = \frac{N_{Re}}{2}\sqrt{\bar{f}/2} \quad ,$$

y θ, es una función empírica de $\bar{y}^+$:

$$\theta = \frac{\bar{y}^+ - 60}{22} \quad .$$

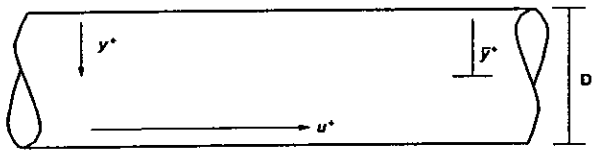


Figura 5.11. DISTRIBUCIÓN DE LA VELOCIDAD DE UN FLUIDO.

El número de Reynolds, N, y el factor de fricción de Fanning, $\bar{f}$, son parámetros adimensionales que dependen de las propiedades físicas y velocidad del fluido, así como de las dimensiones y rugosidad de la tubería. La fórmula para el cálculo del número de Reynolds, es:

$$N_{Re} = \frac{vD\rho}{\mu} \quad ,$$

donde *v* es la velocidad media del fluido, D es el diámetro interno de la tubería, y ρ y μ son la densidad y la viscosidad del fluido, respectivamente; todas estas variables en unidades compatibles.

Para tuberías circulares, no muy rugosas, donde el rango del número de Reynolds varía de 2000 a 50000, $\bar{f}$ puede representarse adecuadamente con la ecuación de Blasius, o sea:

$$\bar{f} = 0.079(N_{Re})^{-0.25} \quad .$$

Realice un programa para calcular la integral

$$\int_a^b f(x)\,dx \quad ,$$

en base a las cuadraturas de Gauss-Chebyshev y de Gauss-Laguerre, para "n" puntos de expansión. Sustituya apropiadamente f(x) y los intervalos, [a, b], en la expresión de la velocidad adimensional, u^+. Compare los resultados logrados con ambas cuadraturas, para N_{Re} = 5,000, 10,000, 25,000 y 50,000, con y^+ = 1, 2, 5, 10, 20, 50, 100, 200, 500 y 1,000; usando 2, 4, 6 10 y 15 puntos de expansión. Grafique en papel semi-logarítmico los resultados de log y^+ vs. u^+, obtenidos para 15 puntos y N_{Re} = 50,000. Compare dicha gráfica con la presentada por Gill y Scher, y redacte comentarios al respecto.


FAC. DE DERECHO * PROFESIONALES
* SECRETARIA DE EXAMENES

APROXIMACIÓN A LA SOLUCIÓN DE ECUACIONES DIFERENCIALES

VI

VI.1 INTRODUCCIÓN.

Aún cuando la mayoría de las ecuaciones diferenciales se resuelven al aplicar técnicas analíticas, existen casos que representan fenómenos físicos que no tienen solución analítica o que la tienen pero es demasiada compleja. Afortunadamente, dichos casos pueden solucionarse mediante aproximaciones numéricas.

Por medio de ecuaciones diferenciales ordinarias puede describirse el comportamiento de innumerables procesos físicos, particularmente aquellos que están en función del tiempo (fenómenos transitorios). Por lo consiguiente, los métodos que dan solución a tales ecuaciones cobran gran importancia tanto para ingenieros como para científicos. En Ingeniería Petrolera, el empleo de ecuaciones diferenciales parciales, se enmarca principalmente en el área de Caracterización de Yacimientos (consultar sección III.2.5) Este capítulo está destinado a la descripción de las técnicas numéricas más usuales, para obtener la solución de ecuaciones diferenciales ordinarias y parciales.

VI.2 ECUACIONES DIFERENCIALES ORDINARIAS.

Una ecuación diferencial ordinaria (EDO), de n-ésimo orden, puede representarse de la siguiente forma:

$$F\left[x, y, \frac{dy}{dx}, \frac{d^2y}{dx^2}, \frac{d^3y}{dx^3}, \ldots\ldots, \frac{d^ny}{dx^n}\right] = 0 \quad . \tag{6.1}$$

La ecuación (6.1) es denominada de *n-ésimo* orden porque la máxima derivada es de orden "n", y *ordinaria* puesto que aparecen únicamente las derivadas totales (no muestra derivadas parciales o posee sólo una variable independiente, x). Una función y(x) que satisfaga esta ecuación, la cual debe ser cuando menos "n" veces diferenciable, será la solución. Para obtener una solución única (en general existen muchas funciones y(x) que satisfacen la ecuación (6.1)), es necesario contar con información adicional, es decir, valores de y(x) o de sus derivadas para algunos valores específicos de x; para una expresión de n-ésimo orden, "n" condiciones, de esa naturaleza, son suficientes para lograr una solución única, y(x). Si las "n" condiciones están referidas para el mismo valor de x (x_0, por ejemplo), entonces se tiene una *EDO con valores o condiciones iniciales.* Cuando se involucra la variable independiente x, el problema se denomina como *EDO con valores o condiciones en la frontera.*

Una EDO de n-ésimo orden puede representarse por un sistema de "n" ecuaciones de primer orden, definiéndose "n-1" nuevas variables. Considérese la siguiente expresión de segundo orden (ecuación de Bessel):

$$x^2 \frac{d^2y}{dx^2} + x\frac{dy}{dx} + \left(x^2 - p^2\right)y = 0 \quad , \tag{6.2}$$

Donde p es una constante. Ahora, introduciendo una nueva variable, z = dy/dx, la expresión de segundo orden puede reescribirse como un par de ecuaciones de primer orden, o sea:

$$\frac{dy}{dx} - z = 0 \quad , \tag{6.3}$$

$$x^2 \frac{dz}{dx} + xz + \left(x^2 - p^2\right)y = 0 \quad . \tag{6.4}$$

Como la gran mayoría de las EDO de orden alto pueden ser manejadas de manera análoga, se desarrollará solamente la solución numérica de ecuaciones de primer orden.

Una EDO de primer grado puede representarse de manera simbólica, de la siguiente manera:

$$y' = f(y, t) \quad , \tag{6.5}$$

la cual tiene una familia de curvas, y(t), como solución. Si f(y, t) = y, para cualquier constante C, la función $y(t) = Ce^{-t}$ es la solución. Al elegir un valor inicial, digamos y(0), se selecciona una curva de la familia, como se observa en la Figura 6.1. El valor inicial de la variable dependiente puede ser especificado para cualquier valor, t_0, de la variable independiente.

Como algunas EDO no tienen solución analítica, es usual emplear métodos numéricos para obtener una aproximación de la solución. Para esto, se requiere de la siguiente información:

1) Una expresión que regule el error tolerable en la solución.
2) Una expresión que indique la dificultad existente para lograr tal solución. Esto, normalmente, está en función del error tolerable.

Las aproximaciones que se desarrollarán en este capítulo, incluyen el uso de métodos Paso a Paso (también conocidos como métodos de Diferencias o de Variables Discretas). En estas técnicas se genera una serie de puntos $t_0, t_1, t_2, \ldots\ldots$, con un intervalo o paso, variable, o sea:

$$h_n = t_{n+1} - t_n \quad . \tag{6.6}$$

En cada punto t_n, la solución, $y(t_n)$, se aproxima por un número y_n, que se calcula a partir de valores previamente obtenidos. Un método de Diferencias que provee una regla para calcular y_{n+1}, usando "k" valores antecedentes $y_n, y_{n+1}, \ldots\ldots, y_{n-k+1}$, es nombrado un método de *k pasos*. Si k = 1, será un método de Pasos Simples o Sencillos; y si k > 1, se denominará método Multipasos.

El método de Euler es un ejemplo de método de pasos simples, y el valor de y_{n+1} se calcula mediante una extrapolación lineal del valor anterior, y_n. Sea, por ejemplo, la siguiente EDO:

$$y' = f(y, t) \quad , \tag{6.7}$$

con condición inicial $y(t_0) = y_0$. La pendiente de la solución puede obtenerse para la condición inicial con la expresión $y_0' = f(y_o, t_0)$. Por lo que, una aproximación de $y_1 = y(t_1)$, se obtiene usando los primeros dos términos de la serie de Taylor, esto es:

$$y(t_1) \approx y_1 = y_0 + h_0 f(y_0, t_0) \quad , \tag{6.8}$$

análogamente para $t_2 = t_1 + h_1$, se tiene:

$$y(t_2) \approx y_2 = y_1 + h_1 f(y_1, t_1) \quad , \tag{6.9}$$

y de manera general:

$$y(t_{n+1}) \approx y_{n+1} = y_n + h_n f(y_n, f_n) \quad . \tag{6.10}$$

A cada paso, la aproximación de la solución cruza de una curva de la familia solución a otra (Figura 6.2). Esta situación provoca errores considerables en la solución de algunas EDO. En el caso de la EDO $y' = y$ (Figura 6.1), el método de Euler magnificaría el error a razón de e^t, mientras "t" se incrementa. Este fenómeno se conoce como *inestabilidad de una ecuación diferencial.*

Por otro lado, si la EDO es $y' = -y$, se tiene la familia de curvas mostradas en la Figura 6.3. Para dicha ecuación, los errores, decrementarán a medida que "t" disminuye. Esto se denomina *estabilidad de una ecuación diferencial.* Recapitulando, para la ecuación diferencial $y' = \lambda y$, donde λ es una constante, el error en la solución se verá afectado por el factor $e^{\lambda t}$, en tanto que "t" aumente. Si $\lambda \leq 0$, los errores no se magnificarán puesto que la ecuación se comportará establemente. Es conveniente señalar que una EDO estable, no necesariamente tiene una expresión numérica sencilla.

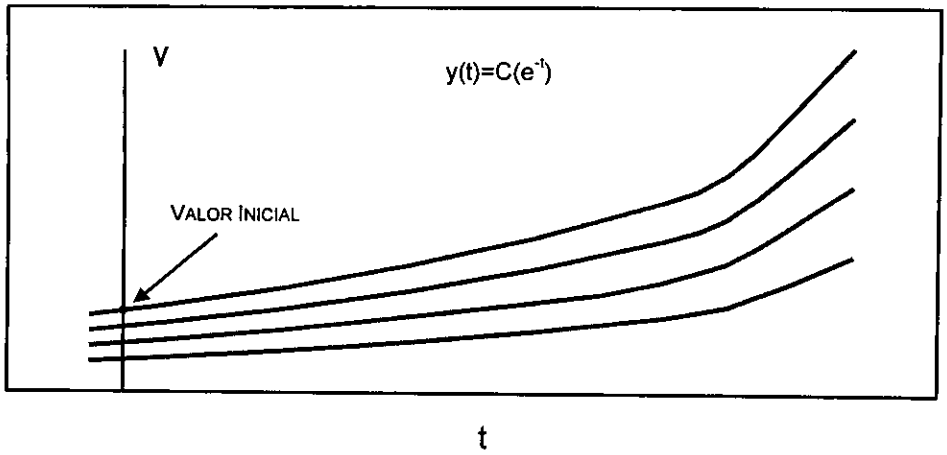


Figura 6.1. FAMILIA DE SOLUCIONES DE Y´ = Y.

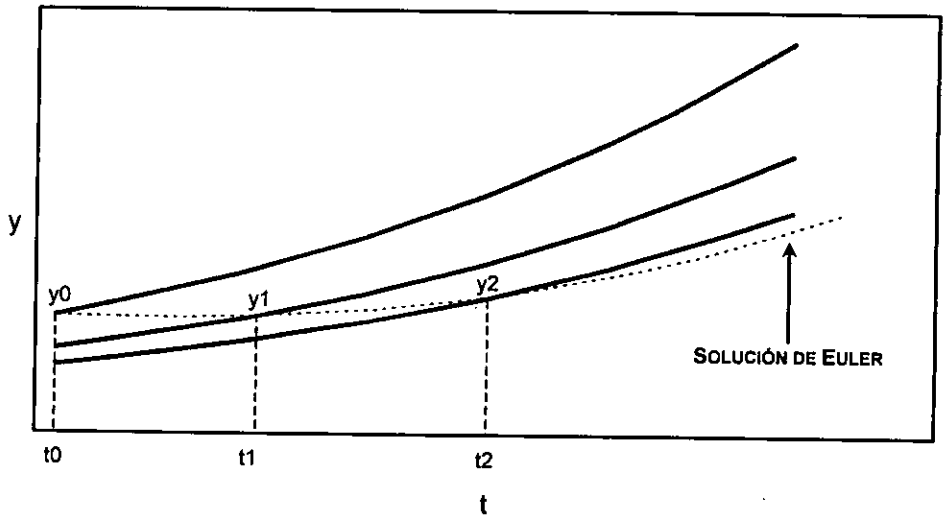


Figura 6.2. MÉTODO DE EULER.

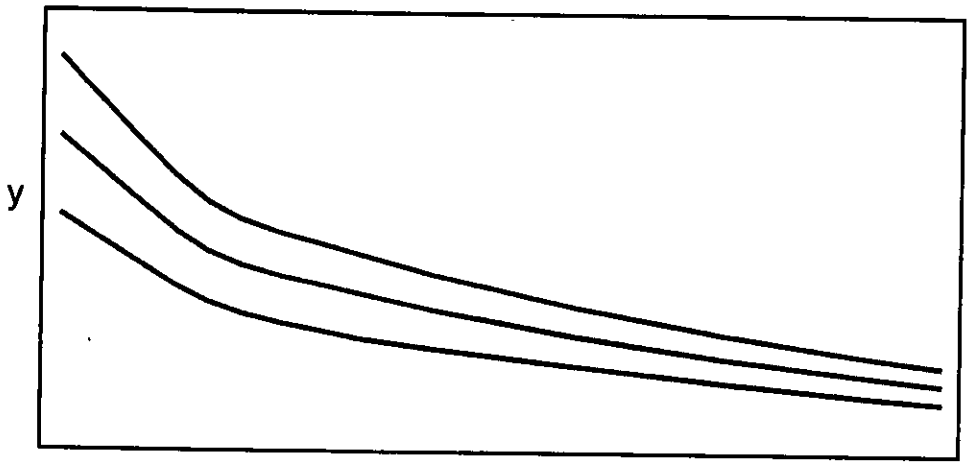


Figura 6.3. FAMILIA DE SOLUCIONES DE Y´ = -Y.

Los errores presentes en la aproximación de la solución de una EDO, son:

1) Error de redondeo.
2) Error de discretización.

El error de redondeo es inherente al equipo de cómputo (consultar sección II.4) y desafortunadamente es impredecible.

El error de discretización depende, exclusivamente, de la técnica empleada. Entonces, si todos los cálculos aritméticos se realizaran con precisión infinita, el único error presente sería éste. Se le subdividide en:

- Discretización local.
- Discretización global.

El error de discretización local se genera en un paso, si el valor previo es exacto y no existe error de redondeo. Para ser más claros, supóngase que $u_n(t)$ es una función en "t", definida por:

$$u_n' = f(u_n, t) \quad , \tag{6.11}$$

$$u_n(t_n) = y_n \quad . \tag{6.12}$$

De esta manera, $u_n(t)$ es la solución de la EDO, determinada a partir del valor t_n y no para la condición inicial original, t_0. Por lo cual, el error de discretización local, d_n, se define como la diferencia entre las soluciones, calculada y teórica, para un mismo valor t_n, o sea:

$$d_n = y_{n+1} - u_n(t_{n+1}) \quad . \tag{6.13}$$

El error de discretización global es la diferencia entre la solución calculada (ignorando el redondeo), y la solución verdadera determinada para el valor t_0. Esto es:

$$e_n = y_n - y(t_n) \quad . \tag{6.14}$$

Para una EDO, donde f(y, t) dependa de "y", el error en la solución para cualquier punto t_n, está regido por las soluciones previamente calculadas. Como consecuencia, el error global será más grande que la sumatoria de los errores locales si la EDO es inestable, y será menor que dicha sumatoria si la EDO es estable. La estabilidad de una EDO, y por lo consiguiente el efecto de los errores local y global, está gobernada por el signo de $\partial f/\partial y$. Para ecuaciones no lineales, $\partial f/\partial y$ puede alternar su signo, por lo que, la ecuación será inestable en algunas regiones y estable en otras. Para sistemas de EDO no lineales, la situación es aún más compleja.

Un concepto primordial en la evaluación de la exactitud de un método numérico, es el *orden*. El orden se define en términos del error de discretización local, obtenido cuando se aplica alguna técnica a problemas con solución sencilla. Se dice que un método es de orden "p", si existe un número "C", tal que:

$$|d_n| \le C\, h_n^{p+1} \quad . \tag{6.15}$$

El número "C" depende de la derivada de la función que define la ecuación diferencial y de la longitud del intervalo sobre el cual se obtuvo la solución, y será independiente del número y del tamaño del paso, "n" y h_n respectivamente.

Una expresión para calcular el error total es:

$$\text{Error Total} \approx b\left(C\,h + \frac{\varepsilon}{h}\right) \quad , \tag{6.16}$$

donde:

$b = t_f - t_0$

t_f = punto fijo final del intervalo de evaluación.

ε = error de redondeo.

El error total puede minimizarse, de manera aproximada, para el valor óptimo de "h", a través de la fórmula siguiente:

$$h_{\text{ópt}} \approx \sqrt{\varepsilon / C} \quad . \tag{6.17}$$

Finalmente, debe indicarse que conforme "h" disminuye, el error de discretización, normalmente también disminuirá, y la solución numérica comenzará a converger con respecto a la verdadera solución. No obstante, cuando el valor de "h" es excesivamente pequeño, la solución numérica divergirá a causa del error de redondeo.

VI.2.1 TÉCNICAS PARA LA SOLUCIÓN NUMÉRICA DE UNA ECUACIÓN DIFERENCIAL ORDINARIA.

A pesar de que existen múltiples métodos para la aproximación de la solución de una EDO, se expondrán los tres más importantes.

VI.2.1.1 MÉTODO DE TAYLOR.

Si y(t) es la solución de una EDO, su expansión en términos de la serie de Taylor es:

$$y(t+h) = y(t) + hy'(t) + \frac{h^2}{2!}y''(t) + \ldots\ldots \quad . \tag{6.18}$$

El método de Euler puede interpretarse como una aproximación al de Taylor (considerando sólo los dos primeros términos de la expansión). Si "y" tiene derivadas de alto orden, una técnica de orden "p", estará dada por:

$$y_{n+1} = y_n + hy_n' + \frac{h^2}{2!}y_n'' + \frac{h^3}{3!}y_n''' + \ldots\ldots + \frac{h^p}{p!}y_n^{(p)} \quad . \tag{6.19}$$

El primer término en despreciarse provee una estimación del error local de discretización y puede usarse para seleccionar el tamaño del paso (intervalo). Las derivadas y'(x), y''(x), etcétera, pueden expresarse en términos de las derivadas parciales de la función, f. Si se

hace esto, debe distinguirse la función f(y, t) con dos variables independientes, de la función f(y(t), t), que posee sólo una variable independiente, obtenida al sustituir la solución "y" en f. Iniciando con:

$$y'(t) = f(y(t), t) \quad , \tag{6.20}$$

se pueden diferenciar ambos lados con respecto a "t", resultando:

$$y'' = f_y y' + f_t = f_y f + f_t \quad , \tag{6.21}$$

entonces, para la tercera diferencial:

$$y''' = f_{yy} f^2 + f_y^2 f + f_y f_t + 2 f_{ty} f + f_{tt} \quad . \tag{6.22}$$

El proceso anterior es denominado *diferenciación total*. Por ejemplo, si $f(x, t) = y^2 + t^2$, se tendrá:

$$y' = y^2 + t^2 \quad , \tag{6.23}$$

$$y'' = 2y^3 + 2yt^2 + 2t \quad , \tag{6.24}$$

$$y''' = 6y^4 + 8y^2 t^2 + 4yt + 2t^4 + 2 \quad . \tag{6.25}$$

Nótese que aún cuando la función f es un polinomio de dos variables, sus derivadas totales se tornan más complicadas conforme incrementa el orden. En la práctica, este método tiene una aplicación muy limitada.

VI.2.1.2 MÉTODO DE RUNGE-KUTTA.

Esta técnica está diseñada para requerir solamente la evaluación de la primera derivada, lo cual representa una ventaja sobre el método de Taylor. La aproximación se logra evaluando varias veces la función f(y, t). La expresión del método de Runge-Kutta de 4°. orden, es:

$$y_{n+1} = y_n + \frac{1}{6}(k_0 + 2k_1 + 2k_2 + k_3) \quad , \tag{6.26}$$

donde:

$$k_0 = h\,f(y_n, t_n) \ , \tag{6.27}$$

$$k_1 = h\,f\left(y_n + \frac{1}{2}k_0, t_n + \frac{h}{2}\right) \ , \tag{6.28}$$

$$k_2 = h\,f\left(y_n + \frac{1}{2}k_1, t_n + \frac{h}{2}\right) \ , \tag{6.29}$$

$$k_3 = h\,f(y_n + k_2, t_n + h) \ . \tag{6.30}$$

Debe resaltarse que en cada paso de cálculo, se realizan cuatro evaluaciones de la función f(y, t). El método de Runge-Kutta es una extensión de la cuadratura de Simpson para EDO, esto es:

$$\int_{t_n}^{t_{n+1}} f(t)\,dt \approx \frac{h_n}{6}\left[f(t_n) + 4f\left(\frac{t_n + t_{n+1}}{2}\right) + f(t_{n+1})\right] \ . \tag{6.31}$$

Si f(y, t) es función exclusiva de "t", ambas técnicas serán equivalentes.

Como en las técnicas de Runge-Kutta no es factible obtener el error local de discretización, se imposibilita la selección del tamaño del intervalo, sin embargo, son fácilmente programables y numéricamente estables para casi toda EDO, además, puede iniciarse *por sí mismo*, puesto que sólo necesita una solución inicial y_0, para calcular y_{n+1}. En cualquier momento del cálculo de la aproximación, el tamaño del intervalo h_n es susceptible de modificarse (la subrutina RKF45, descrita en la sección VI.2.1.5, incluye una moderna modificación que permite el control del tamaño h_n).

VI.2.1.3 MÉTODO MULTIPASOS.

En los métodos previamente expuestos, y_{n+1} se calcula a partir de una función que depende de y_n, t_n y h_n. Es coherente pensar que empleando los valores de y_{n-1}, y_{n-2},....., y f_{n-1}, f_{n-2},...., donde $f_i = f(y_i, t_i)$, la aproximación obtenida sería más exacta. Los métodos Multipasos se basan en esta premisa y son sumamente efectivos. Para soluciones con gran exactitud, requieren menor cantidad de cálculos que las técnicas de Un-paso.

La fórmula general del método Multipasos (k-pasos) Lineal, es la siguiente:

$$y_{n+1} = \sum_{i=0}^{k} \alpha_i y_{n+1-i} + h \sum_{i=0}^{k} \beta_i f_{n+1-i} \quad , \tag{6.32}$$

donde "k" es un número entero, y tanto α_i como β_i son distintos de cero. La técnica se denomina lineal pues cada f_i se comporta linealmente, aunque, f puede no ser una función lineal de sus argumentos.

Una vez que el método ha iniciado, cada paso subsecuente requiere del cálculo de y_{n+1}, a partir de los valores, previamente obtenidos o conocidos, y_n, y_{n-1},.....,y_{n-k+1}, f_n, f_{n-1},.....,f_{n-k+1}. Si $\beta_0 = 0$, la técnica se denomina explícita. Cuando $\beta_0 \neq 0$, la técnica será implícita, ya que debe calcularse f_{n+1} para obtener y_{n+1}.

Un método Multipasos Predictor-Corrector, emplea una técnica explícita (predictor), seguida por una o más aplicaciones de una implícita (corrector). El método de Adams de 4°. orden es un buen ejemplo de esta técnica, sus ecuaciones son:

Ecuación Predictora

$$y_{n+1} = y_n + \frac{h}{24}(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}) \quad , \tag{6.33}$$

Ecuación Correctora

$$y_{n+1} = y_n + \frac{h}{24}(9f_{n+1} + 19f_n - 5f_{n-1} + f_{n-2}) \quad . \tag{6.34}$$

El algoritmo para el método Predictor-Corrector es como sigue:

1) Use la ecuación predictora para calcular $y_{n+1}^{(0)}$, como una aproximación inicial de y_{n+1}. Considere i = 0.
2) Obténgase y evalúese la derivada de la función, o sea:

$$f_{n+1}^{(i)} = f\left(y_{n+1}^{(i)}, t_{n+1}\right) \quad . \tag{6.35}$$

3) Calcule una mejor aproximación $y_{n+1}^{(i+1)}$ usando la ecuación correctora, para:

$$f_{n+1} = f_{n+1}^{(i)} \quad . \tag{6.36}$$

4) Si $\left|y_{n+1}^{(i+1)} - y_{n+1}^{(i)}\right|$ > tolerancia, incremente "i" en una unidad y regrese al punto 2; en otro caso, $y_{n+1} = y_{n+1}^{(i+1)}$, y el problema se da por resuelto.

En este algoritmo se ejecutan tres procesos independientes: el paso predictor (1), denominado P; el paso evaluador (2), llamado E; y el paso corrector (3) o paso C. El 4°. paso puede sustituirse por un ciclo de "m" iteraciones correctoras. El esquema resultante se expresa en forma compacta de la siguiente forma:

$$P(EC)^m \quad , \tag{6.37}$$

por lo que si se consideran la estabilidad y el error de discretización, el método Predictor-Corrector de Adams, más sensible, será[1]:

$$PECE \quad . \tag{6.38}$$

VI.2.1.4 PROBLEMA CON VALORES EN LA FRONTERA.

Hasta aquí se han expuesto EDO con valores iniciales, en las cuales la variable dependiente está disponible para algún valor de la variable independiente, "t". En ocasiones es necesario resolver EDO, sujetas a los valores de la variable dependiente, obtenidos para diferentes valores de "t". Un problema de esa naturaleza es conocido como *problema con valores en la frontera*. Las técnicas para dar solución a estos problemas son totalmente disímbolas de aquellas para problemas con valor inicial.

A continuación se expondrá un método para reducir un problema con valores en la frontera a una secuencia de problemas con valor inicial. Supóngase el sistema de EDO mostrado enseguida:

$$y' = f(y, t) \ , \tag{6.39}$$

donde:

$$y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \ . \tag{6.40}$$

Sujeto a las condiciones $y_1(0) = \alpha$ y $y_2(b) = \beta$, $b \neq 0$.

Para solucionar tal sistema, se sugiere el siguiente método iterativo:

1) Elija un valor ξ que aproxime el valor de $y_2(0)$.

2) Resuelva el problema de valor inicial:

$$\hat{y} = f(\hat{y}, t) \ , \tag{6.41}$$

$$\hat{y}(0) = \begin{pmatrix} \alpha \\ \xi \end{pmatrix} \ . \tag{6.42}$$

3) Si $|\hat{y}_2(b) - \beta| \leq$ tolerancia, entonces $y(t) = \hat{y}(t)$; de otra forma ajuste ξ y retorne al paso 2.

Como información adicional, se comentará que el valor de ξ, está íntimamente relacionado con el concepto de región de convergencia, del método de Newton-Raphson para sistemas de ecuaciones no lineales.

VI.2.1.5 SUBRUTINA RKF45.

RKF45 es una subrutina, basada en la técnica de Runge-Kutta, para resolver una EDO con condiciones iniciales. Se requiere evaluar la función involucrada en seis ocasiones por cada paso de cálculo (para esto debe adicionarse un subprograma a modo de función). Cuatro de esos valores funcionales se combinan con un juego de coeficientes (propuestos por E. Fehlberg[1]), para generar un método de 4°. orden, y uno de 5°. orden se desarrolla al conjugar los seis valores funcionales con otro grupo de coeficientes. La comparación entre ambas cantidades proporciona una aproximación del error, valor con el cual se logra el control del tamaño del intervalo o paso, h_n. Como ya se indicó, las fórmulas usadas por RKF45 necesitan seis valores de la función involucrada por paso, k_1 a k_2. Ellos están definidos por:

$$k_i = h_n f\left(y_n + \sum_{j=1}^{i-1} \beta_{ij} k_j, t_n + \alpha_i h_n \right) \quad ; \quad i = 1, 2, ..., 6 \quad . \tag{6.43}$$

El nuevo valor y_{n+1} se obtiene usando una combinación ponderada de las 6 cantidades k_i, es decir:

$$y_{n+1} = y_n + \sum_{i=1}^{6} \gamma_i k_i \quad . \tag{6.44}$$

Existen 27 coeficientes en total, 6 para α_i, 15 para β_i y 6 para γ_i. La determinación de dichos coeficientes se logra al expandirse todas las k_i en series de Taylor, las expansiones se sustituyen en la fórmula para y_{n+1}, y el resultado se comparar con la serie de Taylor para la solución real (definida en la sección VI.2), $u_n(t_{n+1})$. Fehlberg también encontró un juego de coeficientes γ_i^* que combinados con 4 de los 6 valores de k_i, producen una cantidad y_{n+1}^*, la cual brinda mayor precisión a la técnica de 4°. orden. Esto, es:

$$y_{n+1}^* = y_n + \sum_{i=1}^{6} \gamma_i^* k_i \quad , \tag{6.45}$$

RKF45 no calcula y^*_{n+1}, en su lugar, calcula una estimación del error, $\sum_{i=1}^{6}(\gamma_i - \gamma_i^*)k_i$, la que sirve para regular el tamaño del paso, h_n.

Como corolario, se dirá que RKF45 es la implementación, de propósito general, más efectiva del método de Runge-Kutta, además, es fácil de entender y usar. Por si esto fuera poco, para demandas modestas de precisión (error de discretización local menor de 10^{-5} ó 10^{-6}), RKF45 es tan eficiente como la más sofisticada técnica de Adams. En el CD de consulta se presenta, codificada en lenguaje *Visual Basic*, la subrutina RKF45 así como un programa fuente para su utilización (Rkf45.bas, consultar Apéndice A).

VI.3 ECUACIONES DIFERENCIALES PARCIALES.

Dentro del conjunto de las ecuaciones diferenciales parciales (EDP), las de 2° orden y forma parabólica revisten mayor interés para el Ingeniero Petrolero.

Las EDP de 2° orden, se clasifican como de tipo elíptico, hiperbólico o parabólico.

$$Lu = A\frac{\partial^2 u}{\partial x \partial x} + 2B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y \partial y} + D\frac{\partial u}{\partial x} + E\frac{\partial u}{\partial y} + Fu = \Phi\,, \qquad (6.46)$$

donde las variables independientes "x", "y" pueden intercambiarse a "x", "t". A,...,F, Φ son funciones dadas de "x" y "y".

Las condiciones de frontera lineales son de la forma:

$$B(u) = \alpha u + \beta u_n = f\ ,$$

donde u_n implica $\frac{\partial u}{\partial n}$ en la dirección normal hacia fuera de la frontera. Las cantidades α, β y f pueden variar a lo largo de la frontera.

Si:

$$B^2 - AC \begin{cases} < 0 \rightarrow \text{Operador elíptico.} \\ = 0 \rightarrow \text{Operador parabólico.} \\ > 0 \rightarrow \text{Operador hiperbólico.} \end{cases}$$

Es muy común que una de las variables independientes sea el tiempo, "t", y que las restantes sean coordenadas de distancia, "x", "y", y "z". Enseguida se desarrollará el método de Diferencias Finitas, que es el más "socorrido" para dar solución a una EDP.

VI.3.1 MÉTODO DE DIFERENCIAS FINITAS.

La idea básica de cualquier método de aproximación, consiste en reemplazar el problema original por otro que sea más fácil de resolver, y cuya solución se aproxime a la del problema original.

El Método de Diferencias Finitas (MDF), tiene como principio fundamental el sustituir las derivadas parciales, que intervienen en la EDP de un problema en particular, por su aproximación en Diferencias Finitas. Considérese la siguiente ecuación:

$$\frac{\partial u}{\partial x} = \lim_{\Delta x \to 0} \frac{\Delta u}{\Delta x} = \frac{\Delta u}{\Delta x} \quad . \tag{6.47}$$

En su aproximación, el valor asignado a Δx dependerá de cada problema. El MDF ejemplificado anteriormente, permite transformar un problema de Cálculo Diferencial en uno algebraico que será más fácil de solucionar.

El pilar fundamental en el desarrollo del MDF lo constituye la serie de Taylor. La expansión de f(x) en términos de tal serie, con respecto al punto "x" y con incrementos positivos, puede expresarse como:

$$f(x - \Delta x) = f(x) - \Delta x \frac{df}{dx} - \frac{\Delta x^2}{2!} \frac{d^2 f}{dx^2} - \ldots\ldots - \frac{\Delta x^n}{n!} \frac{d^n f}{dx^n} \quad , \tag{6.48}$$

y con incrementos negativos, se convierte en:

$$f(x+\Delta x)=f(x)-\Delta x\frac{df}{dx}+\frac{\Delta x^2}{2!}\frac{d^2f}{dx^2}-.....+\frac{\Delta x^n}{n!}\frac{d^nf}{dx^n} \quad . \tag{6.49}$$

Empleando las dos expresiones desarrolladas, pueden derivarse aproximaciones en Diferencias Finitas para la primera y segunda derivadas de f(x). Al despejar df/dx de la ecuación (6.48), se obtiene:

$$\frac{df}{dx}=\frac{f(x+\Delta x)-f(x)}{\Delta x}-\frac{1}{\Delta x}\left(\frac{\Delta x^2}{2!}\frac{d^2f}{dx^2}+.....+\frac{\Delta x^n}{n!}\frac{d^nf}{dx^n}\right) \quad . \tag{6.50}$$

Asumiéndose que Δx es pequeño, los términos dentro del paréntesis pueden despreciarse (tienen valores mínimos), además, si se denotan con $\theta(x)$, la ecuación (6.50) se transforma en:

$$\frac{df}{dx}=\frac{f(x+\Delta x)-f(x)}{\Delta x}+\theta(\Delta x) \quad . \tag{6.51}$$

El término $\theta(\Delta x)$ se lee como "de orden Δx", pues precisamente Δx es la parte principal del primer término, es decir, el más significativo (el de mayor valor), de entre todos los que conforman la serie que lo define.

En conclusión, al sustituir la derivada de la función f(x) por $(f(x+\Delta x) - f(x))/\Delta x$, se cometerá un error de truncamiento del orden Δx, $\theta(\Delta x)$.

Otra aproximación de df/dx, se obtiene a partir de la ecuación (6.49), esto es:

$$\frac{df}{dx}=\frac{f(x)-f(x-\Delta x)}{\Delta x} \text{ con un error de orden } \theta(\Delta x) \quad . \tag{6.52}$$

Una tercera aproximación, con error de truncamiento menor, puede ser formulada al restar de la expresión (6.48) la (6.49), o sea:

$$\frac{df}{dx} = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} + \theta\left((\Delta x)^2\right) \quad . \tag{6.53}$$

Estas tres aproximaciones se conocen como Diferencias hacia delante ecuación (6.51), Diferencias hacia Atrás ecuación (6.52) y Diferencias Centrales ecuación (6.53).

Por otro lado, si se suman las expresiones (6.48) y (6.49), se obtendrá una aproximación para la segunda derivada:

$$\frac{d^2f}{dx^2} = \frac{f(x+\Delta x) - 2f(x) + f(x+\Delta x)}{(\Delta x)^2} + \theta\left((\Delta x)^2\right) \quad . \tag{6.54}$$

En caso de derivadas parciales, la aproximación puede obtenerse directamente de las ecuaciones (6.51), (6.52), (6.53) y (6.54); por ejemplo, para la segunda derivada parcial, se tiene:

$$\frac{\partial^2 f}{\partial x^2} = \frac{f(x+\Delta x) - 2f(x) + f(x+\Delta x)}{(\Delta x)^2} + \theta\left((\Delta x)^2\right) \quad . \tag{6.55}$$

Realizando un procedimiento similar, pueden aproximarse derivadas de mayor grado.

VI.3.1.1 ESQUEMA EXPLÍCITO.

Hasta esta parte puede concluirse que el MDF facilita la solución numérica de ecuaciones diferenciales, revisemos ahora, la metodología de su aplicación. Supóngase la siguiente EDP con sus respectivas condiciones iniciales y de frontera:

$$\frac{\partial^2 p}{\partial x^2} = \frac{\partial p}{\partial t} \quad , \tag{6.56}$$

$$\left.\begin{aligned} p(0,t) &= 0 \\ p(1,t) &= 0 \end{aligned}\right\} \text{Condiciones de Frontera} \quad ,$$
$$p(x,0) = 0 \quad \text{Condición Inicial} \quad .$$

El problema consiste en calcular la presión a cualquier distancia, x, y tiempo, t, de interés. Sustituyendo las derivadas parciales por sus respectivas diferencias finitas, se tendrá:

$$\frac{p_{i+1}^n - 2p_i^n + p_{i-1}^n}{(\Delta x)^2} + \theta\left((\Delta x)^2\right) = \frac{p_i^{n+1} - p_i^n}{\Delta t} + \theta(\Delta t) \quad . \tag{6.57}$$

Nótese que $f(x+\Delta x)$ se sustituyó, para manejo de nomenclatura, por p_{i+1}, donde los subíndices manejan distancia y los superíndices tiempo. El superíndice "n" representa el tiempo actual y el "n+1" el tiempo al cual se predice la presión. Si los términos $\theta((\Delta x)^2)$ y $\theta(\Delta t)$ se desprecian, la aproximación de la EDP se reduce a:

$$\frac{p_{i+1}^n - 2p_i^n + p_{i-1}^n}{(\Delta x)^2} = \frac{p_i^{n+1} - p_i^n}{\Delta t} \quad . \tag{6.58}$$

VI.3.1.2 ERRORES DE CONVERGENCIA Y ESTABILIDAD EN EL ESQUEMA EXPLÍCITO.

El error en la aproximación numérica puede definirse como la diferencia entre la solución exacta, p_i^n, y la calculada, Q_i^n, o sea:

$$e_i^n = p_i^n - Q_i^n \quad . \tag{6.59}$$

Con la finalidad de cuantificar la magnitud de tal error y de definir los errores de convergencia, manipularemos algebraicamente algunas de las expresiones anteriores. Restando la ecuación (6.58) de la (6.57), se obtiene:

$$\frac{e_i^{n+1} - e_i^n}{\Delta t} = \frac{e_{i+1}^n - 2e_i^n + e_{i-1}^n}{(\Delta x)^2} + \theta\left((\Delta x)^2 + (\Delta t)\right) \ , \tag{6.60}$$

haciendo $r = \Delta t/(\Delta x)^2$ y despejando a e_i^{n+1}:

$$e_i^{n+1} = re_{i+1}^n + (1+2r)e_i^n + re_{i-1}^n + \theta\left((\Delta t)^2 + \Delta t(\Delta x)^2\right) \ . \tag{6.61}$$

Ahora, asignando valores absolutos en ambos miembros de la expresión anterior y considerando que "r" está definida en el intervalo [0, ½], se tiene que:

$$\left|e_i^{n+1}\right| \le r\left|e_{i+1}^n\right| + (1+2r)\left|e_i^n\right| + r\left|e_{i-1}^n\right| + c\left((\Delta t)^2 + \Delta t(\Delta x)^2\right) \ , \tag{6.62}$$

Donde el elemento $c((\Delta t)^2 + \Delta t(\Delta x)^2)$ resulta de la definición siguiente: Un término A se denomina de orden $(\Delta x)^n$ o $\theta(\Delta x)^n$, si existe una constante, c, positiva e independiente de Δx, tal que $A < c(\Delta x)^n$. Si esto se cumple, tanto (Δx) como A tenderán a cero, al menos con la misma velocidad.

Por otro lado, el máximo error al tiempo "n" puede estimarse con la siguiente expresión:

$$\left|e^n\right| = \text{máx}\left|e_i^n\right| \quad , \quad i = 0, 1, ..., n \ , \tag{6.63}$$

entonces, tomando en cuenta el máximo error, la ecuación (6.62) se reduce a:

$$\left|e^{n+1}\right| \le \text{máx}\left|e^n\right| + c\left((\Delta t)^2 + \Delta t(\Delta x^2)\right) \ . \tag{6.64}$$

Reduciendo un paso en el tiempo, la desigualdad anterior puede transformarse en:

$$\left|e^n\right| \le \text{máx}\left|e^{n-1}\right| + c\left((\Delta t)^2 + \Delta t(\Delta x^2)\right) \ . \tag{6.65}$$

Misma que al sustituirse en la expresión (6.64), resulta:

$$\left|e^{n+1}\right| \le \left|e^{n-1}\right| + 2c\left((\Delta t)^2 + \Delta t\left(\Delta x^2\right)\right) \ , \tag{6.66}$$

continuando con reducciones sucesivas en el tiempo, se llega a:

$$\left|e^{n+1}\right| \le \left|e^{0}\right| + mc\left((\Delta t)^2 + \Delta t\left(\Delta x^2\right)\right) \ , \tag{6.67}$$

además, como $e^0 = 0$, por las condiciones iniciales y de frontera, el máximo error al tiempo "n+1", queda definido por la relación:

$$\left|e^{n+1}\right| \le mc\left((\Delta t)^2 + \Delta t\left(\Delta x^2\right)\right) \ . \tag{6.68}$$

Se dice que un esquema en diferencias finitas es convergente, si en un punto (x, t) se cumple que:

$$\lim_{\substack{\Delta x \to 0 \\ \Delta t \to 0}} \left(p_{i,j} - Q_{i,j}\right) = 0 \ . \tag{6.69}$$

Con base en la expresión (6.68) y en la definición (6.69), se concluye que el esquema explícito es convergente si $r \le ½$, es decir, $(\Delta x)^2 > 2\Delta t$. En otras palabras, si se restringe el cociente $\Delta t/(\Delta x)^2$ a valores menores de ½, la presión pronosticada poseerá un grado de exactitud aceptable.

Otro punto relevante en la aplicación del MDF, lo constituye el error de estabilidad. Un sistema es inestable si el error aumenta conforme se incrementa el tiempo; esto, en notación algebraica, es:

$$\frac{e^{n+1}}{e^n} \leq 1 \Rightarrow \text{Sistema Estable} \quad , \tag{6.70}$$

$$\frac{e^{n+1}}{e^n} > 1 \Rightarrow \text{Sistema Inestable} \quad . \tag{6.71}$$

Recapitulando, la ecuación (6.58) es un desarrollo explícito, en diferencias finitas, de la expresión (6.56), condicionalmente convergente y estable, para valores de $\Delta t/(\Delta x) < ½$. El esquema se denomina *explícito*, ya que es posible el despeje de la única incógnita $\left(p_i^{n+1}\right)$, esto es:

$$p_i^{n+1} = \frac{\Delta t}{(\Delta x)^2}\left(p_{i+1}^n - 2p_i^n + p_{i-1}^n\right) + p_i^n \quad ; \quad i = 1, 2, \ldots\ldots, n \quad . \tag{6.72}$$

Para emplear la ecuación (6.72) en un caso práctico, es necesario contemplar las restricciones establecidas para Δx y Δt.

Por ejemplo, puede ser conveniente elegir incrementos pequeños en el espacio, sin embargo, ello conlleva a la utilización de incrementos en el tiempo más pequeños aún, con el consecuente incremento del tiempo de cómputo; y esto no es deseable, sobre todo cuando el equipo electrónico usado es rentado o "prestado".

VI.3.1.3 ESQUEMA IMPLÍCITO.

Si en la ecuación (6.56) la derivada espacial (∂x) es sustituida por diferencias finitas al tiempo "n+1", y no al tiempo "n" como en el desarrollo explícito, y la derivada temporal (∂t) es utilizada con un incremento hacia atrás, relativo al tiempo "n+1", se realiza un desarrollo implícito. Analíticamente se tiene:

$$\frac{p_{i+1}^{n+1} - 2p_i^{n+1} + p_{i-1}^{n+1}}{(\Delta x)^2} + \theta\left((\Delta x)^2\right) = \frac{p_i^{n+1} - p_i^n}{\Delta t} + \theta(\Delta t) \quad . \tag{6.73}$$

El término *implícito*, surge a raíz de que en este esquema no es posible despejar (explicitar) las incógnitas. Agrupando elementos y haciendo $r = \Delta t/(\Delta x)^2$, la ecuación (6.73) queda como sigue:

$$p_{i+1}^{n+1} - \left(2 + \left(1/r\right)\right) p_i^{n+1} + p_{i-1}^{n+1} = -\left(1/r\right) p_i^n \quad ; \quad i = 1, 2, \ldots\ldots, n \quad . \tag{6.74}$$

La expresión (6.74) representa un sistema tridiagonal de "n" ecuaciones. Si $\alpha = 2 + (1/r)$, el citado sistema, en forma matricial, es:

$$\begin{bmatrix} -\alpha & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -\alpha & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -\alpha & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -\alpha & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -\alpha & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -\alpha \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ \vdots \\ q_{n-1} \\ q_n \end{bmatrix}^{n+1} = -\frac{1}{r} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ \vdots \\ q_{n-1} \\ q_n \end{bmatrix}^{n} \quad . \tag{6.75}$$

Obsérvese que en la derivación de este sistema se consideran las siguientes condiciones de frontera:

$$p_0^{n+1} = p_{n+1}^{n+1} = p_0^n = p_{n+1}^n = 0 \quad . \tag{6.76}$$

VI.3.1.4 ERRORES DE CONVERGENCIA Y ESTABILIDAD EN EL ESQUEMA IMPLÍCITO.

El error máximo de este esquema puede obtenerse mediante la siguiente expresión:

$$e = \frac{1}{1 + r\left[2 - \cos\left(\Delta x\left(\frac{n\pi}{L}\right)\right)\right]} \quad , \tag{6.77}$$

donde "n" es el número de celdas y "L" es la longitud total en estudio, además, $|e| \leq 1$ para todo valor de "r", lo cual asegura la estabilidad del Esquema Implícito.

En base al Teorema de Equivalencia de Lax ("Dado un problema de valor inicial propiamente definido, y dada su aproximación por diferencias finitas, estabilidad es la condición necesaria y suficiente para la convergencia"), este esquema es convergente. Por lo tanto, el Esquema Implícito, representado por la ecuación (6.74), es incondicionalmente estable y convergente, ya que el error de aproximación no depende de Δx y/o de Δt.

VI.3.1.5 ESQUEMAS PONDERADOS.

Un esquema de solución más general se obtiene al considerar familias de diferencias finitas a través de promedios ponderados. El esquema en diferencias finitas de la EDP (6.56), puede expresarse como:

$$\frac{\theta\partial^2 x\, p_i^{n+1} + (1-\theta)\partial^2 x\, p_i^n}{(\Delta x)^2} = \frac{p_i^{n+1} - p_i^n}{\Delta t} \quad , \tag{6.78}$$

donde:

$$\partial^2 x\, p_i^n = p_{i+1}^n - 2p_i^n + p_{i-1}^n \quad . \tag{6.79}$$

En la ecuación (6.78), el error es del orden de $\theta((\Delta x)^2 + (\Delta t)^2)$. Debe observarse que el esquema es explícito si $\theta = 0$ y totalmente implícito si $\theta = 1$.

VI.3.1.6 ERRORES DE CONVERGENCIA Y ESTABILIDAD EN EL ESQUEMA PONDERADO.

El esquema ponderado es condicionalmente estable y convergente de acuerdo con:

$$r \le \frac{1}{2(1-2\theta)} \quad , \quad 0 \le \theta \le \frac{1}{2} \quad , \tag{6.80}$$

e incondicionalmente estable y convergente para valores dentro del rango $½ \le \theta \le 1$. En la práctica, los valores de θ más empleados son 0, ½ y 1. En el caso particular de que $\theta = ½$, el esquema implícito incondicional conformado es denominado como Esquema de Crank-Nicholson.

En resumen, con los esquemas implícitos se logra mejorar las condiciones de estabilidad, al introducir sistemas incondicionales, y obtener mayor exactitud al disminuir el orden del error.

VI.3.1.7 TIPOS DE CONDICIONES DE FRONTERA.

Cuando las condiciones de frontera, como en el caso estudiado, la EDP se clasifica como de tipo *Dirichlet* o de primera clase. Si tales condiciones están dadas en términos de derivadas parciales de la función incógnita, la EDP se clasifica como de tipo *Neumann* o de segunda clase. La combinación de estas condiciones de frontera genera una tercera clasificación, las EDP de tipo *Robin* o de tercera clase.

VI.3.1.8 ECUACIONES DIFERENCIALES PARCIALES EN DOS DIMENSIONES.

El desarrollo de la aproximación de una EDP en dos dimensiones, se discute ampliamente en el capítulo III, sección III.2.5, "Solución al problema de flujo bidireccional, monofásico, transitorio, a través de medios porosos".

VI.4 PROBLEMAS RESUELTOS.

Problema Resuelto VI.1. La función error está definida por la integral mostrada enseguida:

$$\operatorname{erf}(x)=\frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}\,dt \quad ,$$

pero también puede expresarse como la solución de una ecuación diferencial[1], esto es:

$$y'(x)=\frac{2}{\sqrt{\pi}}e^{-x^2} \quad ,$$

cuya condición inicial es y(0) = 0. Realice un programa de cómputo que empleando la subrutina RKF45, despliegue una tabla de erf(x) para x = 0.0, 0.1, 0.2,...,0.9, 1.0.

Compare sus resultados con los obtenidos en el Capítulo II, sección II.6, problema resuelto II.1, y con los arrojados usando la subrutina Quanc8 (Capítulo V, sección V.5.1).

Solución:

El programa elaborado se presenta en el de consulta (RKF45.exe, consultar el Apéndice A). Algunos resultados se consignan en la Tabla 6.1. Se observa que presentan un error relativo máximo del 25%, con respecto a los valores tabulados de erf(x). El tiempo total de ejecución del programa fue de 1.05 segundos.

x	erf(x)				
	Subrutina RKF45	Serie de Taylor	Artificio Forsythe	Subrutina Quanc8	Valores Tabulados
0.0	0.0	0.0	0.0	0.0	0.0
0.5	0.5204999	0.5205002	0.5204999	0.515584	0.52049
1.0	0.8427008	0.8426992	0.8427007	0.832869	0.8413

Tabla 6.1. RESULTADOS DEL CÁLCULO DE erf(x).

VI.5 PROBLEMAS PROPUESTOS.

Problema Propuesto VI.1. M. Muskat desarrolló un procedimiento para predecir el comportamiento de yacimientos homogéneos, con distribución uniforme de presión, sin entrada de agua y sin segregación gravitacional de fluidos. La ecuación de dicho método es diferencial ordinaria de primer grado, y expresa la variación de la saturación de aceite a cualquier presión*:

$$\frac{ds_o}{dp} = \frac{s_o\left(X_p + \left(k_g/k_o\right)Y_p\right) - Z_p s_g}{1 + \left(k_g/k_o\right)\left(\mu_o/\mu_g\right)},$$

* GARAICOCHEA P., F. Y BASHBUSH B., J. L., "APUNTES DE COMPORTAMIENTO DE LOS YACIMIENTOS", FACULTAD DE INGENIERÍA, U.N.A.M., 1987.

donde:

$$X_p = \frac{B_g}{B_o}\frac{dR_s}{dp},$$

$$Y_p = \frac{\mu_o}{\mu_g}\frac{1}{B_o}\frac{dB_g}{dp},$$

$$Z_p = \frac{1}{B_g}\frac{dB_g}{dp}$$

Elabore un programa de cómputo que empleando RKF45 dé solución a estas ecuaciones.

TÉCNICAS AVANZADAS DE PROGRAMACIÓN CON APLICACIÓN A LA INGENIERÍA PETROLERA

VII

VII.1 INTRODUCCIÓN.

El objetivo de este capítulo, es proporcionar algunos conceptos de uso cotidiano en la programación de computadoras personales dentro de la Industria Petrolera.

En la primera parte se presentan los programas de cómputo SIMPP y REGRESML, que son útiles en la simulación del comportamiento de presión en pruebas de decremento y en el análisis de regresión lineal, respectivamente.

El examen minucioso de estos programas permitirá obtener un conocimiento de las técnicas básicas para el manejo de "menús interactivos", ayudas de uso de programa, graficación en pantalla, métodos estadísticos, y otros tópicos de interés, tales como el empleo de una antitransformada del plano de Laplace al plano Real para resolver ecuaciones diferenciales parciales (programa SIMPP).

Se incluyen conceptos asociados con el "diseño óptimo de programas de cómputo", los cuales se consideran de gran utilidad para el desarrollo adecuado de programas computacionales.

VII.2 PROGRAMAS REGRESML Y SIMPP.

El programa REGRESML contiene un modelo de regresión múltiple, que apoyado en técnicas de álgebra matricial, permite realizar ajustes de ecuaciones a conjuntos de datos usando el método de Mínimos Cuadrados, además, mediante procedimientos estadísticos se puede evaluar la *bondad* de dicho ajuste.

Frecuentemente, en el área de Ingeniería Petrolera se presentan problemas en los cuales dos o más variables están relacionadas y es necesario investigar la relación matemática que las une. REGRESML calcula los coeficientes del modelo de regresión propuesto por el usuario a un conjunto de datos, también efectúa cálculos estadísticos que incluyen análisis de varianza. Es posible realizar gráficas de las estadísticas de regresión para analizar en forma rápida el ajuste del modelo matemático al conjunto de datos.

Antes de predecir a partir de un modelo de regresión múltiple, es necesario tener la certeza de que los parámetros calculados para el modelo sean estadísticamente significativos. Un parámetro no significativo debe suponerse igual a cero y eliminarse. Pueden emplearse varias pruebas estadísticas si los errores están normalmente distribuidos, tales como la "F de Fisher" o la "t de Student" (las cuales están programadas en REGRESML). Es decir, no se busca únicamente una función matemática qué nos diga de que manera están relacionadas las variables, sino que se desea saber con qué precisión se puede predecir el valor de la variable dependiente en función de variables independientes conocidas. Las técnicas utilizadas para lograr este objetivo se conocen como *Métodos de Correlación*.

Los métodos de regresión se usan para determinar la mejor relación funcional entre las variables, mientras que los métodos de correlación se utilizan para medir el grado de asociación de las distintas variables.

REGRESML presenta un *ambiente conversacional*, y ofrece: ayudas que pueden invocarse con la tecla F1, resultados en forma de tablas y gráficas (en pantalla), y maneja la ejecución del programa mediante opciones que se despliegan en forma de menús. El análisis detallado

de este programa facilita la selección de procedimientos que posteriormente puedan ser utilizados como subrutinas en otros proyectos.

En lo referente al programa SIMPP, éste constituye un simulador de presión que permite calcular pruebas de decremento en base a información de las características de la formación productora, del fluido producido y del tipo de terminación del pozo. El programa utiliza el *algoritmo de Sthefest* para realizar la antitransformada de Laplace (del plano de Laplace al Real), de la solución de la ecuación diferencial que caracteriza el flujo radial laminar, en un yacimiento homogéneo e isótropo, de espesor constante, conteniendo un fluido ligeramente compresible, con flujo isotérmico, considerando el efecto de almacenamiento de pozo y el factor de daño, esto es[19]:

$$L\{p_{wD}\} = \frac{K_0\left(\sqrt{p}\right) + s\sqrt{p}\,K_1\left(\sqrt{p}\right)}{p\left[\sqrt{p}\,K_1\left(\sqrt{p}\right) + \bar{C}_p\left\{K_0\left(\sqrt{p}\right) + s\sqrt{p}\,K_1\left(\sqrt{p}\right)\right\}\right]}\ , \qquad (7.1)$$

donde:

L = Transformada de Laplace de un argumento.
p_{wD} = Caída de presión adimensional.
K_0 = Función Bessel modificada de segunda clase.
K_1 = Función Bessel modificada de primera clase.
s = Factor de daño.
p = Variable de la transformada de Laplace.
$\bar{C}_p$ = Coeficiente adimensional de almacenamiento del pozo.

Las técnicas para realizar antitransformadas de Laplace se usan con mucha frecuencia debido a que es mucho más fácil obtener las soluciones de Ecuaciones Diferenciales en el plano de Laplace, y además, a que no siempre se tienen soluciones manejables computacionalmente en el plano Real; en el caso de la solución programada en SIMPP se requiere menos de un minuto para obtener la simulación de una prueba con 30 puntos, pero se emplea más de media hora para lograr el mismo objetivo si se utiliza la solución en el

plano Real, puesto que están involucradas integrales definidas numéricamente (de difícil evaluación).

A partir del uso de SIMPP es posible diseñar eficientemente pruebas de decremento de presión, a fin de lograr el mayor provecho en la caracterización del yacimiento. En el artículo "Diseño de Pruebas de Incremento de Presión"[19], se presentan ejemplos de aplicación de este tipo de simuladores a casos prácticos de campo, por lo cual se recomienda su lectura.

La técnica de Sthefest, de acuerdo con publicaciones especializadas, ha tenido éxito al usarse en el área de pruebas de presión en yacimientos, pero es importante mencionar que esta metodología no siempre reporta buenos resultados con todas las ecuaciones diferenciales, en cuyo caso deberán utilizarse técnicas alternas.

Los diagramas de flujo simplificados de los programas REGRESML y SIMPP, se incluyen en las páginas 215 y 216, y los programas fuente pueden encontrarse en el CD de consulta.

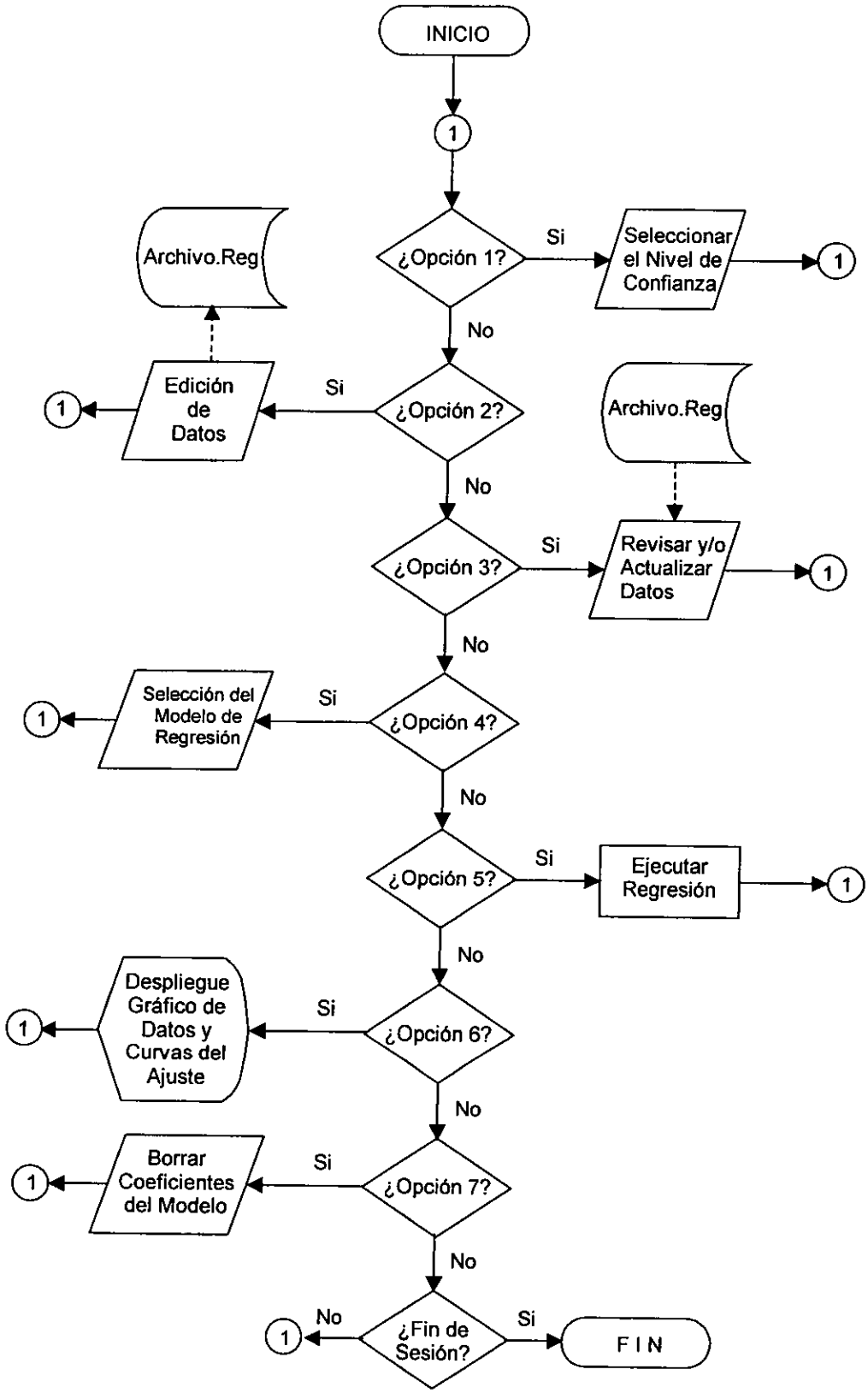


Figura 7.1. DIAGRAMA DE BLOQUES DEL PROGRAMA RESGRESML.

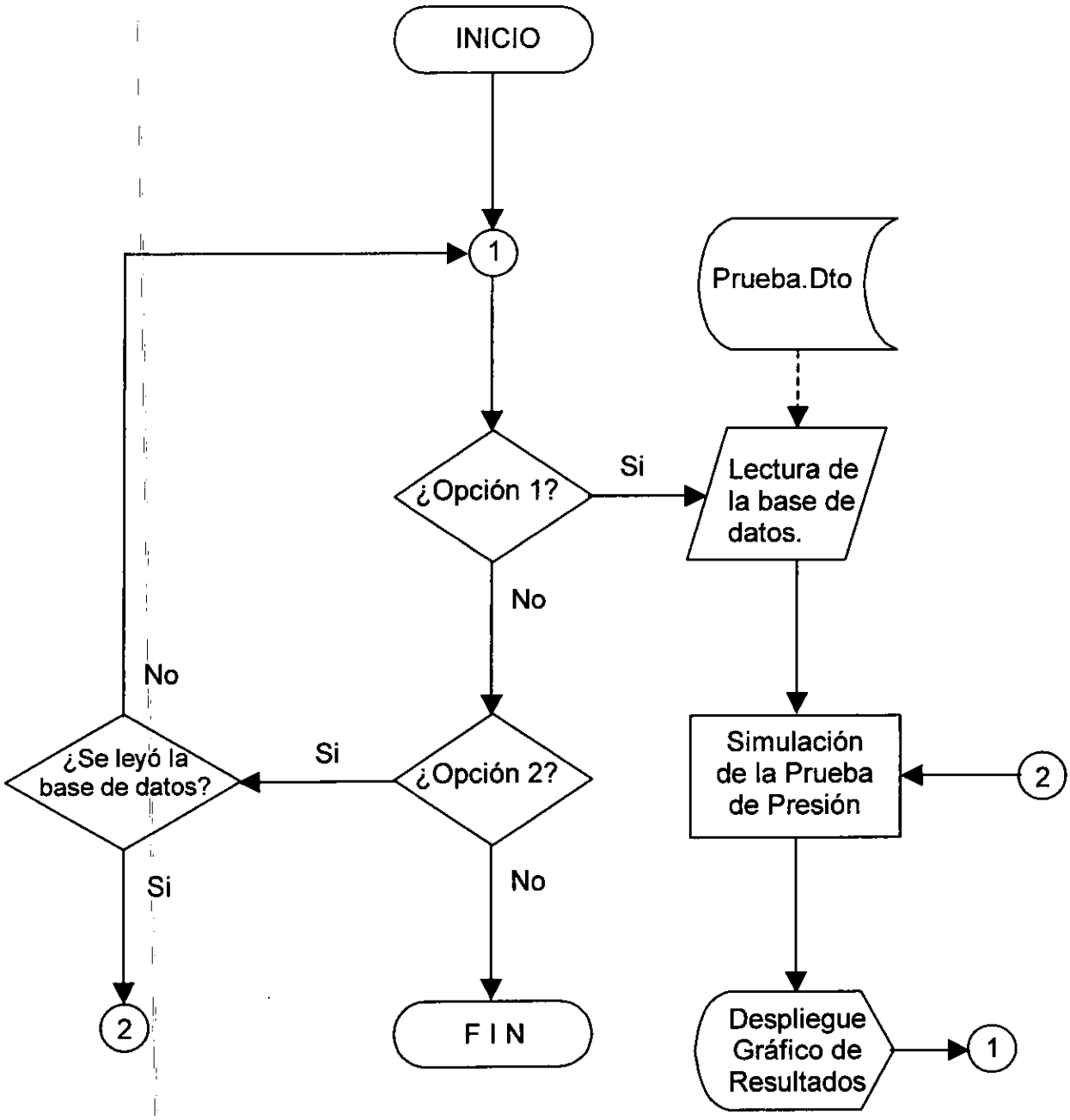


Figura 7.2. DIAGRAMA DE BLOQUES DEL PROGRAMA PARA EL SIMULADOR DE PRUEBAS DE DECREMENTO DE PRESIÓN.

VII.3 DISEÑO ÓPTIMO DE PROGRAMAS DE CÓMPUTO.

El desarrollo de programas de cómputo eficientes, es la meta de todo buen analista. Comparando los tiempos de ejecución de dos programas realizados para el mismo fin, se observa que el tiempo es menor para el programa más eficiente. La rapidez de ejecución de un programa puede afectarse apreciablemente al utilizar códigos de programación (conjunto de instrucciones en *Fortran*, *Visual Basic*, *Pascal*, etcétera) ineficientes. ¿Por qué preocuparnos por el tiempo de ejecución, sí las computadoras realizan los procesos a velocidades muy grandes? Si bien es cierto que los equipos de cómputo modernos son capaces de realizar cientos de operaciones por segundo, también lo es el hecho de que existen problemas ingenieriles cuya solución requiere del manejo de demasiada información y de gran número de operaciones repetitivas.

Un programa eficiente de cómputo debe tomar en cuenta las siguientes características:

- Velocidad de ejecución.
- Entradas y salidas convenientes.
- Facilidad de implementación en otras máquinas.
- Mensajes de diagnóstico a errores cometidos por el usuario.

A continuación se enumeran algunas recomendaciones importantes para la obtención de programas eficientes:

1) USO DE SUBÍNDICES MÚLTIPLES.

Siempre que sea posible use un solo subíndice. Suponga un programa (en lenguaje *Visual Basic*), que calcule la saturación de aceite, s_o, en función de la saturación de gas y agua (s_g y s_w, respectivamente), para cada uno de los bloques en los cuales se ha discretizado un yacimiento (ver Figura 7.3):

```
DIM SW(10,10,10), SO(10,10,10), SG(10,10,10)
FOR I = 1 TO NX
    FOR J = 1 TO NY
        FOR K = 1 TO NZ
            SO(I,J,K) = 1.0 - SG(I,J,K) - SW(I,J,K)
NEXT K,J,I
```

La computadora almacena la información contenida en cualquier arreglo en forma de vector, esto es:

$$\begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} \qquad \begin{bmatrix} 2 \\ 4 \\ 3 \\ 1 \end{bmatrix}$$

REPRESENTACIÓN EN PAPEL — REPRESENTACIÓN INTERNA EN LA COMPUTADORA

Por tanto, en la ejecución del programa mostrado anteriormente, el compilador traduce a un sólo subíndice, M:

$$M = 100 \times (K-1) + 10 \times (J-1) + I \quad . \tag{7.2}$$

Este cálculo se realiza internamente cada vez que se ejecuta el FOR ... NEXT, lo cual genera 6 multiplicaciones, 6 sumas y 6 restas, para efectuar el proceso en cada celda (cálculo de s_o).

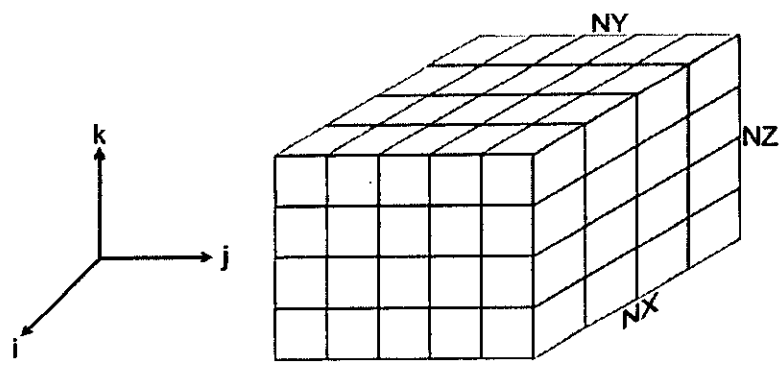


Figura 7.3. YACIMIENTO HIPOTÉTICO EN TRES DIMENSIONES.

El mismo procedimiento puede ser ejecutado como:

```
DIM SW(1000), SO(1000), SG(1000)
M = 0
FOR I = 1 TO NX
    FOR J = 1 TO NY
        FOR K = 1 TO NZ
            M = M + 1
            SO(M) = 1.0 - SG(M) - SW(M)
NEXT K,J,I
```

En el cual sólo se requiere una suma para cada asignación de s_o en una celda.

Otro procedimiento más eficaz, que no requiere de operaciones intermedias es el siguiente:

```
DIM SW(1000), SO(1000), SG(1000)
NXYZ = NX × NY × NZ
FOR M = 1 TO NXYZ
    SO(M) = 1.0 - SG(M) - SW(M)
NEXT M
```

2) MANEJO DE FUNCIONES TABULADAS.

Muchas propiedades en programas de Ingeniería Petrolera se introducen en forma de tablas, por ejemplo propiedades de los fluidos y/o de la roca. El valor de una variable dependiente para su correspondiente variable independiente, se calcula por interpolación de los datos tabulados. Algunos autores han investigado acerca de algoritmos de búsqueda para interpolación, en datos ordenados en forma ascendente, con respecto a su valor numérico, y concluyen que la *Investigación Secuencial* es la más adecuada cuando se tiene un número reducido de datos (10 a 20), mientras que para un número mayor la *Investigación Binaria* es mejor.

La Investigación Secuencial consiste en localizar el intervalo de interés mediante una búsqueda sistemática y ordenada en cada uno de los intervalos entre datos (si se tienen N datos se presentan N-1 intervalos). Un código en *Visual Basic*, para obtener por interpolación un valor Y para un X dado, con Búsqueda Secuencial es:

```
FOR I = 1 TO N-1
      IF X >= X(I) AND X < X(I+1) THEN GOTO 10
NEXT I
10 '*** PROCEDIMIENTO DE INTERPOLACION ***
```

La Investigación Binaria, por otro lado, se basa en dividir el total de datos en dos partes iguales, posteriormente se investiga en que mitad queda el intervalo de interés, el cual se considera ahora como el total de datos. El proceso se repite hasta que se llega al intervalo adecuado. El siguiente programa realiza una Búsqueda Binaria:

```
I = 1
J = N + 1
DO
    K = (I + J)/2
    IF X < X(K) THEN J = K
    IF X >= X(K) THEN I = K
LOOP WHILE X > X(I+1)
'*** PROCEDIMIENTO DE INTERPOLACION ***
```

3) USO DE SUBRUTINAS.

El llamado a una subrutina dentro de un programa requiere de una cierta cantidad de tiempo para búsqueda interna dentro de la memoria de la computadora, el cual se incrementa con el número de argumentos a transferir. Debe evitarse llamar frecuentemente una subrutina (por ejemplo dentro de un FOR .... NEXT), es mejor duplicar el código (conjunto de instrucciones) en esta parte del programa en lugar de llamarlo como subrutina; cuando sea necesario

hacerlo, las variables deben pasarse por COMMON SHARED en lugar de ser transferidas como argumentos.

4) ENTRADAS Y SALIDAS.

Siempre que sea posible debe utilizarse para entrada/salida (input/output, I/O) la opción más apropiada a los datos a manejar. Si los datos van a consultarse o modificarse continuamente (archivos de cuentas corrientes en un banco, *stock* de un supermercado, control de pasajes de una línea aérea, etcétera), la mejor alternativa es emplear Archivos de Acceso Aleatorio o Directo.

Los registros en modo de Acceso Directo proporcionan un acceso a un cierto registro sin tener que los registros anteriores, con lo que el acceso será inmediato (rápido), ahorrando con esto una cantidad considerable de tiempo; el inconveniente es que no se aprovecha eficazmente el espacio en disco, ya que al almacenar registros de menor longitud que la establecida, se malgasta espacio, por ejemplo, si la longitud fija de cada registro es de 256 bytes y se sitúan sólo 125 en cada registro, aproximadamente el 50% de cada registro se convertirá en espacios en blanco.

Visual Basic asigna por defecto una longitud fija de 32,767 bytes, pero el usuario puede definirlo en la opción LEN. Otra desventaja es que se necesita conocer el número que indica la posición del registro, además el proceso de I/O es un tanto difícil de programar.

El siguiente programa es un ejemplo en lenguaje *Visual Basic 6.0* que escribe el archivo "prueba.dat" y que contiene como registros los números del 1 al 10,000, tiene capacidad para leerlo y con opción de cambiar el contenido de un registro en particular. A continuación se presenta el código del problema así como la ventana de resultados.

```
'Escribir registro
Open "c:\mis documentos\prueba.dat" For Random As #1 Len = 80
   For j = 1 To 10000
   Put #1, j, j
   Next j
   j = 100: z = 1000
   Put #1, j, z
   Close #1

'Escribir registro
Open "c:\mis documentos\prueba.dat" For Random As #1 Len = 80
x = Val(Text1.Text)
Get #1, x, y
Text2.Text = y
Label2.Caption = Format("Contenido del registro : ")
Close #1
```

En la figura 7.4 se muestran los resultados obtenidos de haber escrito el archivo "prueba.dat" y el de haber leído el registro "50" el cual contenía el número 50.

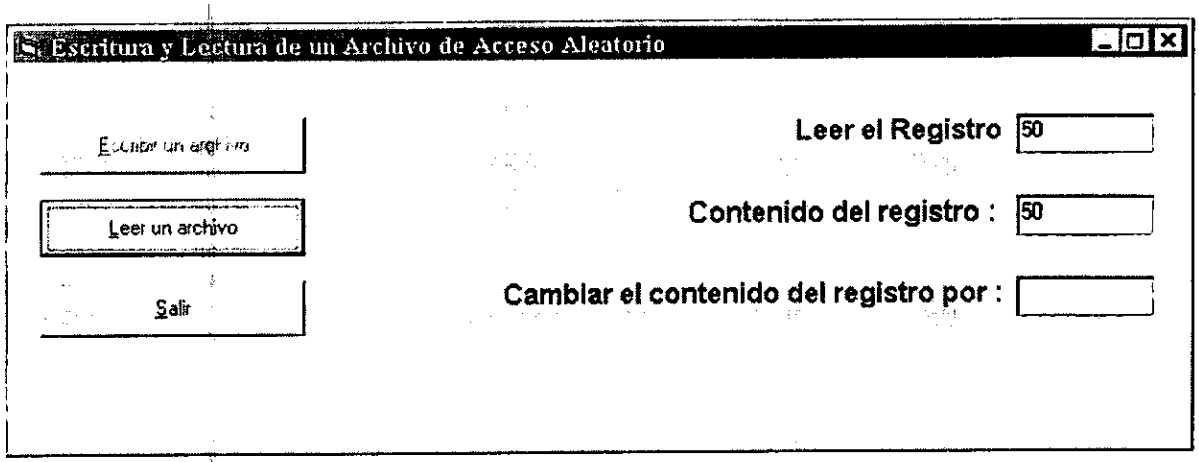


Figura 7.4. RESULTADO DE LA ESCRITURA Y LECTURA DE UN ARCHIVO DE ACCESO ALEATORIO.

En la figura 7.5 se muestran los resultados obtenidos de haber cambiado el contenido del registro "100" por el de 987.

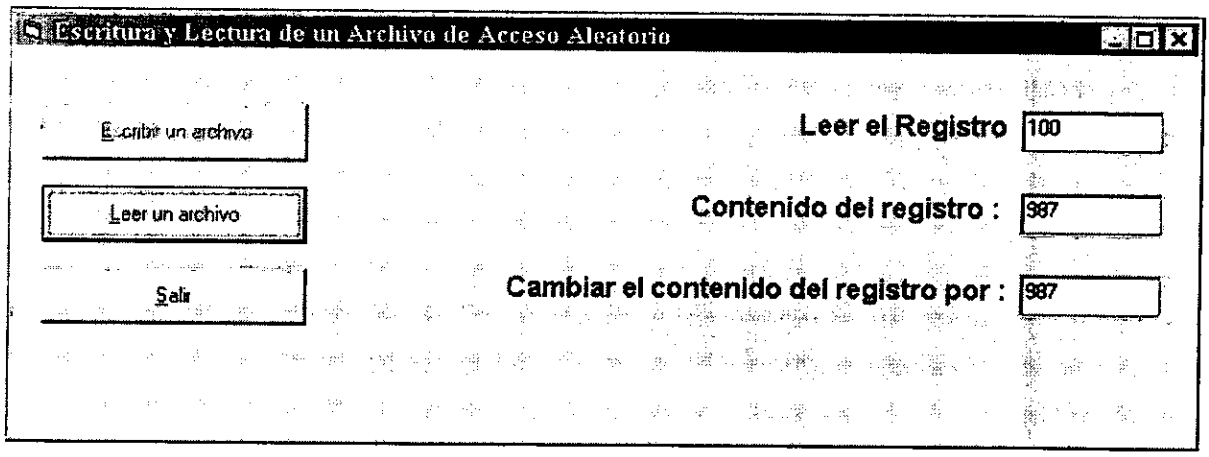


Figura 7.5. RESULTADO DE CAMBIAR UN REGISTRO EN UN ARCHIVO DE ACCESO ALEATORIO.

El siguiente código de programa desarrollado en *Visual Basic 6.0* es un ejemplo de aplicación del área de yacimientos usando este tipo de archivos aleatorios.

```
´Definición de un registro llamado yacimientos
Type yacimientos
    formación as string*15
    porosidad as real
    permeab as real
End type
'Declaración de una variable del tipo yacimientos
Dim roca as yacimientos
'Apertura del archivo para acceso aleatorio
Open "petro.dat" for random as #1 len = len (roca)
'Verificación del numero de registros existente
Reg = lof(1)/len(roca)
```

```
'Proceso de i/o
Roca.formación = val(text1.text) 'formación
Roca.porosidad = val(text2.text) 'porosidad
Roca.permeab = val(text3.text) 'permeabilidad
'Almacenar registro
Put #1, reg, roca
'Leer registro
Get #1, reg, roca
Label1.caption = "formación:" & roca.formación
Label2.caption = "porosidad:" & roca.porosidad
Label3.caption = "permeabilidad:" & roca.permeab
'Cerrar archivo abierto
Close #1
```

Cuando se tiene un número relativamente pequeño de datos con longitudes diversas, se recomienda el uso de Archivos de Acceso Secuencial, tales archivos emplean eficientemente el espacio en disco flexible (su tamaño depende exclusivamente de la longitud y cantidad de cada uno los registros, por lo que no se tienen espacios en blanco innecesarios). La secuencia de I/O es fácil de programar y como se leen todos los registros a la vez, el acceso es más lento comparado con el de los archivos directos. Cabe mencionar que el orden de escritura de variables debe respetarse al efectuar una lectura posterior. A continuación se presenta un procedimiento para manejo de archivos secuenciales:

```
'Almacenar registro
Open "petro.dat" for output as #1
Input "formación"; formación : print #1, formacion
Input "porosidad"; porosidad : print #1, porosidad
Input "permeabilidad"; permeabilidad : print #1, permeab
'Cerrar archivo abierto
Close #1
```

```
'Leer registro
Open "petro.dat" for input as #1
Input #1, formacion : print "formacion :"; formacion
Input #1, porosidad : print "porosidad :"; porosidad
Input #1, permeab : print "permeabilidad :"; permeab
'Cerrar archivo abierto
Close #1
```

Obsérvese que las instrucciones de Input y Print usadas en los archivos de acceso secuencial son cambiadas por las de Put y Get en los archivos de acceso directo

Se puede concluir que el tiempo requerido para ejecutar operaciones de I/O es función del número de *items* (variables) en la lista de entrada/salida.

5) CONSIDERACIONES ADICIONALES.

a) Evite mezclar aritmética (conversiones INTEGER a REAL).
b) En lugar de (2*A) use (A+A).
c) La multiplicación es más rápida que la división, por lo que en vez de (A/2) use (0.5*A)
d) Emplee (A*A) en lugar de (A^2).
e) Evite dejar expresiones constantes dentro de procesos iterativos.
f) Codificar en *lenguaje de Maquina* comparado con el código que utiliza un compilador (*Fortran*, *Visual Basic*, *Pascal*, etcétera), puede incrementar la velocidad de ejecución de 2 a 3 veces.
g) Utilice *almacenamiento (arreglo) dinámico* en lugar de *estático*.

6) FACILIDAD DE IMPLEMENTACIÓN.

a) Evite usar caracteres especiales en los nombres de las variables.

b) Es recomendable usar un máximo de 6 caracteres en los nombres asignados a un variable (recuerde: 1 carácter igual a un 1 byte). De preferencia emplear nombre que identifiquen fácilmente a las variables codificadas.

c) Al capturar un programa de cómputo coloque comentarios alusivos a la instrucció codificada, a fin de que cualquier usuario pueda comprender el proceso, en caso d requerirse su modificación o inclusión como módulo (subrutina) en otro programa.

VII.5 PROBLEMAS PROPUESTOS.

Problema Propuesto VII.1. Modifique el programa REGRESML a fin de obtener una comparación estadística con la F de Fischer,

Problema Propuesto VII.2. Modifique el programa SIMPP a fin de simular pruebas de incremento de presión.

Problema Propuesto VII.3. Simule, empleando el programa SIMPP, una prueba de decremento de presión, considerando:

q_o = 6919 BPD	h = 164 pie
B_o = 2.3	p_i = 8901.72 (lb/pg^2)
k = 50 mD	C_D = 10,000
c_t = 2.5316E-05 (lb/pg^2)$^{-1}$	μ = 0.39 cp
r_w = 0.208 pie	ϕ = 0.02

Calcule la presión para 0.1, 0.2, 0.3, 0.4, 0.5, 1, 2, 4 , 6, 7, 8, 9, 10, 11, 12, 15, 17, 20, 30 y 40 horas. Aplique la técnica semilogarítmica de MDH para corroborar la calidad de los datos de presión calculados.

CONCLUSIONES Y RECOMENDACIONES

VIII

- La asignatura de Programación Avanzada no sólo tiene como objetivo el que los alumnos conozcan y manejen con soltura los diferentes métodos y técnicas de solución numéricas para resolver los innumerables problemas que se presentan en el ámbito petrolero, sino que también aprendan a darles una aplicación práctica; es decir, utilizar los algoritmos ya existentes para resolver algún problema en específico, y así tener un punto de partida para generar nuevas aplicaciones o mejorarlos según sea el caso.

- El presente trabajo es de gran utilidad como apoyo, tanto para el profesor como para el alumno, debido a que posee información importante que puede ser aplicable a diversos problemas de la industria petrolera.

- La programación orientada a objetos parece ser el paradigma de programación dominante en la actualidad, y ha sustituido a las técnicas de programación estructurada que se desarrollaron al principio de los años setenta.

- Visual Basic 6.0 permite un mayor acceso a la potencia y a algunas de las ventajas de la programación orientada a objetos (POO), razón por el cual se decidió utilizarlo para desarrollar todos los programas que incluye esta tesis.

- Las presentaciones en Microsoft PowerPoint incluidas en este trabajo, contienen material adicional de gran utilidad para el profesor en la exposición de la signatura.

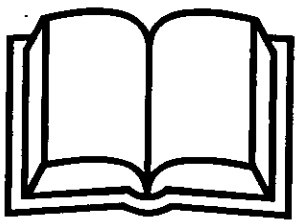
- Con la gran variedad de técnicas, métodos y algoritmos numéricos que existen es necesario tomar en cuenta sus alcances y limitaciones antes de seleccionarlos usando el más adecuado para la resolución de cada problema particular, evitando así operaciones innecesarias, empleando mayor cantidad de memoria RAM o hacer que sea más tardada la convergencia.

- Las subrutinas que aquí se mencionan pueden constituir una herramienta útil para aquellos lectores que en el futuro su trabajo demande el desarrollo de aplicaciones de cómputo ligados a problemas reales más específicos.

- Se requiere emplear tanto el análisis humano como el gran poder de las computadoras y el software, ya que no se conseguirían los resultados óptimos, si sólo se usará el análisis humano o este binomio computadora-software debido a que es uno complemento del otro.

- Es recomendable aprovechar los beneficios que se tiene con la programación orientada a objetos debido a que se cuenta con una herramienta poderosa y fácil de programar ya que los programas desarrollados se podrán utilizar para ser parte de otra aplicación o sólo tomar el módulo de interés.

REFERENCIAS



1. Forsythe, G. E., Malcom, M. A. y Moler, C. B., *Computer Methods for Mathematicals Computations*, Prentice Hall, Inc., Englewodd Cliffs, Nueva jersey, 1977.

2. Carnahan, B., Luther, H. A. y Wilkes, J. D., *Applied Numerical Methods*, John Wiley & Sons, Londres, 1969.

3. Berlanga Gutiérrez, J. M., *Apuntes de Computación Aplicada a la Ingeniería Petrolera*, Facultad de Ingeniería, U.N.A.M., 1980.

4. Borras García, H. E., Iriarte V. Balderrama, R. y Durán Cuevas, R., *Apuntes de Métodos Numéricos*, Facultad de Ingeniería, U.N.A.M., 1985.

5. Solar González, E. y Speziale de Guzmán, L., *Apuntes de Algebra Lineal*, Facultad de Ingeniería, U.N.A.M., 1985.

6. Bitter, Gary. G., *Computación (Fundamentos, Aplicaciones, Programación)*, Addison-Wesley Iberoamericana, 1989.

7. Mano, Morris M., *Arquitectura de Computadores*, Prentice-Hall Hispanoamericana S.A., 1989.

8. Rodríguez Nieto, R. y De la Fuente García, G., *Curso Introductorio de Simulación de Yacimientos*, I.M.P., publicación No. 72 BH/094, 1972.

9. Limón H., J. Tomás, *Transporte de Gas en Régimen Permanente*, I.M.P., publicación No. 17/74 , proyecto D-341A, 1974.

10. Luthe, R., Olivera, A. y Schutz, F., *Métodos Numéricos*, Editorial Limusa, México, 1979.

11. Bourgoyne, A. T. Jr. y Young, F. S. Jr., *A Multiple Regression Approach to Optimal Drilling and Abnormal Pressure Detection*, S.P.E. Journal, Agosto, 1974.

12. Garaicochea Petrirena, F., *Apuntes de Transporte de Hidrocarburos*, Facultad de Ingeniería, U.N.A.M., 1983.

13. Dake, L. P., *Fundamentals of Reservoir Engineering*, Elsevier Publishing Co., Gran Bretaña, 1978.

14. Hernández García, G. y Berlanga Gutiérrez, J. M., *Evaluación de Volúmenes Originales de Hidrocarburos Empleando Métodos Geoestadísticos (Aplicación al Campo Miguel Alemán-Chicontepec),* revista Ingeniería Petrolera, Vol. XXIII, No. 3, marzo de 1983.

15. Hernández García, G., *Curso de Geoestadística*, clave 00246B2P, I.M.P., marzo de 1992.

16. Sthefest, Harald, *Algorithm 368 Numerical Inversion of Laplace Transforms*, Communications of the ACM, Vol. 13, No. 1, enero de 1970.

17. Park, Chan S., *Interactive Microcomputer Graphics*, Addison Wesley Publishing Co., 1985.

18. Agarwal, Ram G., Al-Hussainy, R. y Ramey, H. J. Jr., *An Investigation of Wellbore Storage and Skin Effect in Unsteady Liquid Flow: I. Analytical Treatment*, Transactions of AIME, Vol. 249, 1970.

19. Martínez Romero, N. y León Ventura, R., *Diseño de Pruebas de Incremento de Presión*, XXIV Congreso de la Asociación de Ingenieros Petroleros de México.

20. Thomas Richards McCalla, *Introduction to Numerical Methods and Fortran Programming*, Joho Wiley and Sons, Inc. 1967.

21. S.S. Rao, *Optimization*, Wiley Eastern Limited.

22. Atkinson, K. *An Introduction to Numerical Analysis,* Wiley, New York, 1988.

23. Rodríguez de la Garza F. y Galindo Nava A. P., *Fundamentos de Simulación Numérica de Yacimientos*, Facultad de Ingeniería, División de Estudios de Posgrado, Sección Ingeniería Petrolera, U.N.A.M., 2000.

APÉNDICE

RELACIÓN DE PROGRAM APLI

Los archivos con e
desde el explorador
dto, reg, res y ayd
notas de Windows

ARCH

SUBDIRECTORIO	CONTENIDO (ARCHIVOS Y BASES DE DATOS)
Capítulo IV	Spline.exe Spline.bas Int3dim.exe Pres.dat T-32.dat T-100.dat T-190.dat T-180.dat Svd.exe Svd.bas Optperf.exe Densequi.dto Diengast.dto Pesbarre.dto Presform.dto Profun.dto Reynolds.dto Velpenet.dto Velrotac.dto Kriging.exe Datos.dto
Capítulo V	Quanc8.exe Gaussq.exe m(p).exe mp1.dat mp2.dat mp3.dat
Capítulo VI	Rkf45.exe Rkf45.bas
Capítulo VII	Regresml.exe Archivo.reg Tfile.ayd Ayuda1.ayd Ayuda2.ayd Ayuda3.ayd Simpp.exe Prueba.dto

LÁMINAS PARA APOYO DIDÁCTICO DEL PROFESOR

APÉNDICE B

Las presentaciones que aquí se muestran son una guía importante para que el profesor imparta la clase de Programación Avanzada ya que contienen elementos y conceptos básicos de las subrutinas utilizadas en los diferentes programas, como también los principales proyectos que se generan y que abarcan las tres áreas de la Ingeniería Petrolera; Producción, Perforación y Yacimientos, aunado a esto se presenta una serie de láminas que contienen una introducción al *Visual Basic 6.0* en términos básicos que son fundamentales de conocer para comenzar a programar.

Estas presentaciones son las siguientes:

- Subrutinas Básicas.
- Redes de Gas.
- Regresión Múltiple Aplicada a la Perforación.
- Simulador de un Yacimiento de Gas en 2D.
- Visual Basic 6.0.

A continuación se muestran las láminas de cada una de estas presentaciones antes mencionadas.

Universidad Nacional Autónoma de México

Facultad de Ingeniería, Ingeniería Petrolera

Programación

Avanzada

Subrutinas Básicas

Ariel Guzmán 2001

1

Contenido :

Subrutinas.

- Subrutina Decomp.
- Subrutina Solve.
- Subrutina Spline
- Función Seval.
- Subrutina SVD (Singular Value Descomposition) (Descomposición del Valor Singular, DVS)
- Subrutina Quanc8 (Quadrature Adaptive of Newton-Cotes 8-panel).

Ariel Guzmán 2001

2

Subrutinas : *Decomp y Solve.*

Éste método usa el Pivoteo Parcial que consiste en hacer unos la diagonal principal y dejar ceros en la matriz diagonal inferior para entonces hacer una sustitución hacia atrás. Para esto hay que seleccionar pivotes con valores absolutos grandes que impliquen multiplicaciones con valor absoluto menor o igual a uno, para así disminuir los errores.

Éstas subrutinas se basan en el Método de Eliminación Gaussiana esto es:

A Decomp le corresponde la primera parte de la eliminación; dejar ceros debajo de la diagonal principal.

Solve usa eso cáculos para hacer una sustitución hacia atrás y generar la solución del sistema lineal.

Ariel Guzmán 2001

3

Subrutina : *Decomp.*

Sub Decomp(NDIM, N, a( ), COND, IPVT( ), WORK( ))

DATOS DE ENTRADA:
NDIM: Dimensionamiento del arreglo de la matriz A.
N: Orden de la matriz.
A: Matriz que ser triangularizada.

DATOS DE SALIDA:
A: Contiene una matriz triángular superior (U) y una versión permutada de una triángular inferior (I-L), por lo tanto: (matriz permutada)*A = L*U.

COND: Valor estimado del número de condición de A. Para un sistema lineal A*X = B, cambios en los valores de A y/o B, pueden causar modificaciones a COND y a veces en X. Si COND+1 = COND, A es una matriz singular aún al trabajarla con precisión, entonces COND = 1.0E+32 si la singularidad exacta es detectada.

IPVT: Vector pivote.
IPVT(K): El índice del k-ésimo renglón pivote.
IPVT(N): (-1)^(número de intercambios).

Ariel Guzmán 2001

4

Subrutina : *Decomp.*

ESPACIO DE TRABAJO:

El vector WORK debe declararse e incluirse en la llamada a esta subrutina. Su contenido de entrada se ignora (cero) y el de salida no es relevante.

El determinante de A puede obtenerse mediante la expresión:

Det(A) = IPVT(N) * A(1,1) * A(2,2) * ... * A(N,N)

COND = (1-NORMA de A)*(estimación de 1-NORMA de la inversa de A)

La estimación se obtiene en un paso iterativo inverso para el vector singular reducido. Esto involucra la solución de 2 sistemas de ecuaciones, (TRANSPUESTA de A)*Y = E y A*Z = Y, donde E es un vector de elección (+1 ó -1) para adicionarse en Y.

Estimación = (1-NORMA de Z)/(1-NORMA de Y)

Ariel Guzmán 2001

5

Subrutina : *Solve.*

Sub Solve(NDIM, N, a( ), b( ), IPVT( ))

Solución del sistema lineal, A*X = B. SI la subrutina DECOMP detectó singularidad se omite la ejecución de este procedimiento.

DATOS DE ENTRADA:
NDIM: Dimensionamiento del arreglo de la matriz A.
N: Orden de la matriz.
A: Matriz triángularizada, obtenida por DECOMP.
B: Vector de términos independientes.
IPVT: Vector pivote, obtenido por DECOMP.

DATOS DE SALIDA:
B: Vector solución, X. Representación de un Sistema de Recolección por medio de Nodos y Conectores de Nodos.

Ariel Guzmán 2001

6

Subrutina : *Spline.*

Son funciones cúbicas que ajustan n-1 polinomios de 3er grado; un polinomio para cada una de los n-1 intervalos.

SPLINE calcula los coeficientes de la función cúbica.

Ariel Guzmán 2001
7

Subrutina : *Spline.*

Sub Spline(N, X( ), Y( ), B( ), C( ), D( ))

Los coeficientes B(I), C(I) y D(I), I=1,2,...,N, son calculados para formar el polinomio de Splines cúbicos:

S(X) = Y(I) + B(I)*(X-X(I)) + C(I)*(X-X(I))^2 + D(I)*(X-X(I))^3
para: X(I) =< X =< X(I+1).

DATOS DE ENTRADA:
N: Número de datos, puntos o coordenadas (N >= 2).
X: Abscisa de las coordenadas dato en estricto orden ascendente.
Y: Ordenada de las coordenadas dato.

DATOS DE SALIDA:
B, C, D: vectores con coeficientes de SPLINE, indicados anteriormente.
Usando la letra "P" para denotar diferenciales:
Y(I) = S(X(I))
B(I) = SP(X(I))
C(I) = SPP(X(I))/2
D(I) = SPPP(X(I))/6

Ariel Guzmán 2001
8

Nota: La función SEVAL evalúa el SPLINE.

Función: *Seval.*

Sub Seval(N, U, X( ), Y( ), B( ), C( ), D( ), OP)

Esta subrutina evalúa la función cúbica spline correspondiente al dato a interpolar:

Seval = Y(I) + B(I)*(U-X(I)) + C(I)*(U-X(I))^2 + D(I)*(U-X(I))^3

donde:
X(I) < U < X(I+1), usando la regla de Horner.
Si U < X(1) entoces se supone que I=1.
Si U > X(N) entoces se supone que I=N.

DATOS DE ENTRADA:
N: Número de datos.
U: Abscisa en la que se valuar el SPLINE.
X, Y: Vectores de abscisas y ordenadas dato.
B, C, D: Vectores de coeficientes cálculados por SPLINE.
OP: Variable que rige el tipo de búsqueda, secuencial si **OP=1** y binaria en cualquier otro caso.

Si U no se ubica en el intervalo de la llamada previa, entonces una búsqueda binaria o secuencial ser ejecutada, para determinar el intervalo apropiado.

Ariel Guzmán 2001
9

Subrutina : *Svd (Singular Value Descomposition).*

Sub Svd(NM, M, N, A(), W(), MATU, U(), MATV, V(), IERR, RV1())

Esta rutina determina la descomposición de los valores singulares T, $A=U\Sigma V^t$ de una matriz rectangular de M por N, se emplea una bidiagonalización común y una variante del algoritmo QR.

DATOS DE ENTRADA:

NM: Debe ser el núm. de renglones de los arreglos bidimensionales que se declaran en la sentencia de llamado a la subrutina. Nótese que NM debe ser cuando menos igual al máximo valor entre M y N.

M: Es el número de renglones de A (y de U), esto es, el número de pares de datos base.

N: Es el número de columnas de A (y de U) y el orden de V. Además, representa el grado del polinomio más 1 para la aproximación funcional, es decir, el número de coeficientes de tal función de aproximación.

A: Representa la matriz rectangular de entrada, que ser descompuesta.

MATU: Debe asignarse el valor de 1, si en la descomposición se desea obtener la matriz U, y el de 0 en caso contrario.

MATV: Debe asignarse el valor de 1, si en la descomposición se desea obtener la matriz V, y el de 0 en caso contrario.

Ariel Guzmán 2001
10

Subrutina : *Svd (Singular Value Descomposition).*

DATOS DE SALIDA:

A: Permanece inalterada (a no ser que se reemplace por U o V).

W: Contiene los N valores singulares (no-negativos) de A (elementos de las diagonales de S). Los cuales están desordenados. Si se detecta un error automáticamente se realiza la salida, y los valores singulares se corregirán por los índices IERR+1, IERR+2,..., N.

U: Contiene la matriz U_{mxn} (vectores columna ortogonales) de la descomposición si MATU=1, en caso contrario, se emplea como un arreglo temporal. Puede coincidir con la matriz A. Si se detecta un error, automáticamente se realiza la salida, y las columnas de U multiplicadas por los índices de corrección ofrecen los valores singulares efectivos.

V: Contiene la matriz V_{nxn} (ortogonal) de la descomposición si MATV=1, en caso contrario, no se calcula. Puede coincidir con la matriz A si U no se requiere. Si se detecta un error, automáticamente se realiza la salida, y las columnas de V multiplicadas por los índices de corrección ofrecen los valores singulares efectivos.

IERR: Tendrá el valor:
0: en una ejecución normal.
K : si el k-ésimo valor singular no se ha obtenido después de 30 iteraciones.
RV1: Es un arreglo para almacenaje temporal.

Ariel Guzmán 2001
11

Subrutina : *Quanc8 (Quadrature Adaptive of Newton-Cotes 8-panel).*

Sub Quanc8(A, B, ABSERR, RELERR, RESULT, ERREST, NOFUN%, FLAG)

Esta subrutina estima la integral de la función FUN(X), en el intervalo [A, B] considerando la tolerancia proporcionada por el usuario, y representa una subrutina auto-adaptable basada en la regla de los 8-paneles de Newton-Cotes.

Emplea una técnica de cuadratura adaptada para aproximar el valor de una integral definida.

Variables de tipo real: FUN, A, B, ABSERR, RELERR, RESULT, ERREST, FLAG
Variables de tipo entero: NOFUN

DATOS DE ENTRADA:
FUN: Nombre de la función que valúa la ecuación a integrar en el punto X{ FUN(X) }.
A: El límite inferior de la integración.
B: El límite superior de la integración (B puede ser menor que A).
RELERR: Tolerancia para el error relativo (debe ser positivo).
ABSERR: Tolerancia para el error absoluto (debe ser positivo).

Ariel Guzmán 2001
12

Subrutina : *Quanc8 (Quadrature Adaptive of Newton-Cotes 8-panel).*

DATOS DE SALIDA:

RESULT: Aproximación de la integral que virtualmente satisface el mínimo valor de entre las dos tolerancias de error.

ERREST: Estimación de la magnitud del error actual.

NOFUN: Número de valuaciones de la función (FUN), usadas en el cálculo de RESULT.

FLAG: Factor de integridad. Si FLAG es cero, entonces RESULT probablemente satisfaga la tolerancia de error. Si FLAG es XXX.YYY, entonces XXX ser el número de intervalos que no convergen, y 0.YYY la fracción del último intervalo, a la izquierda, realizado antes de que el valor límite de NOFUN se halla alcanzado.

Variables de tipo real: W0, W1, W2, W3, W4, AREA, X0, F0, STONE, STEPP, COR11, TEMP

Variables de tipo real: QPREV, QNOW, QDIFF, QLEFT, ESTERR, TOLERR

Variables de tipo real: QRIGHT(31), F(16), X(16), FSAVE(8, 30), XSAVE(8, 30)

Variables de tipo entero: LEVMIN, LEVMAX, LEVOUT, NOMAX, NOFIN, LEV, NIM, I, J

Ariel Guzmán 2001

13

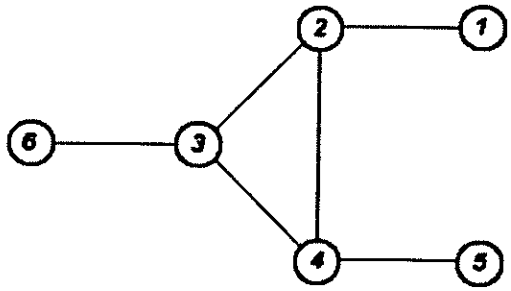
Universidad Nacional Autónoma de México

Facultad de Ingeniería, Ingeniería Petrolera

Programación

Avanzada

Redes de Gas

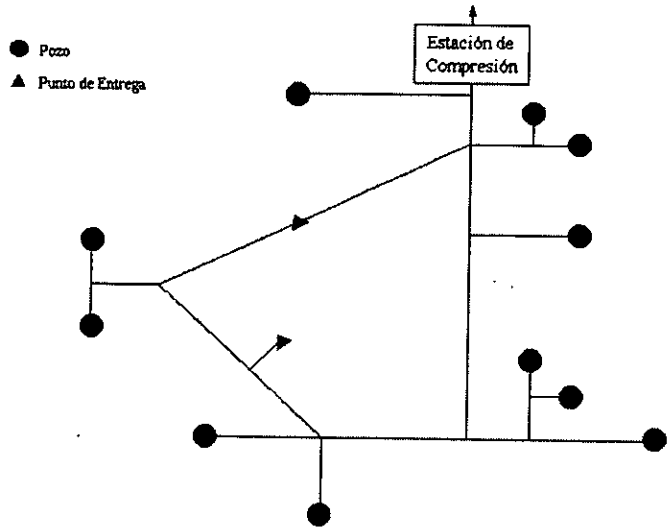


Ariel Guzman
2001
1

Redes de Gas

Programación
Avanzada

Sistema de recolección de gas típico



Ariel Guzman
2001
2

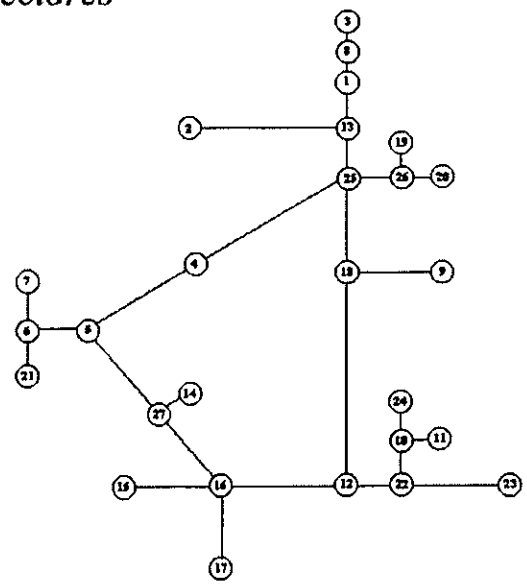
Redes de Gas

Programación Avanzada

Representación de un sistema de recolección por medio de nodos y conectores

○ Nodos

— Conector de Nodos



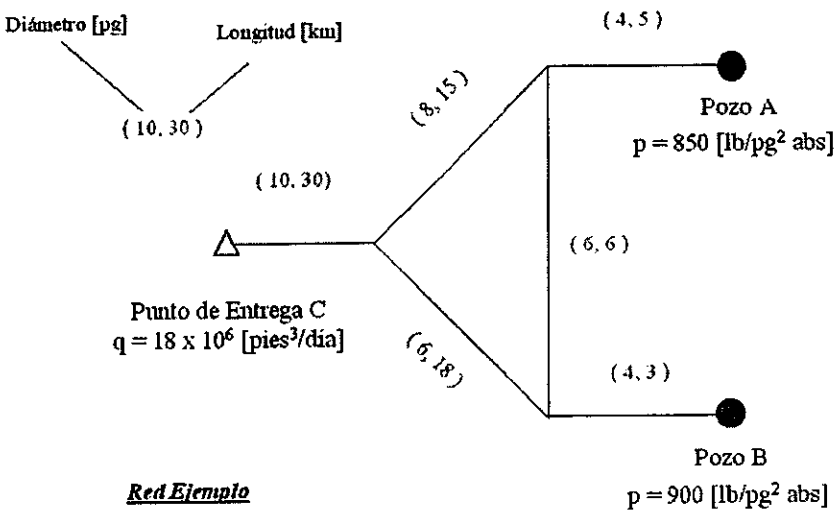
Anel Guzman 2001

3

Redes de Gas

Programación Avanzada

Representación de una red de gas (conectores)



Red Ejemplo

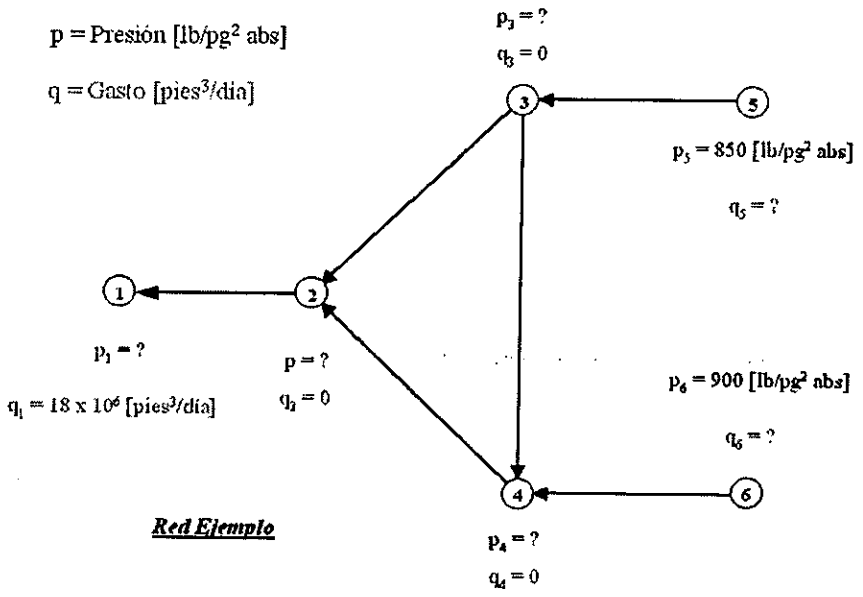
Anel Guzman 2001

4

Redes de Gas

Programación Avanzada

Representación de una red de gas (nodos)



Ariel Guzman 2001

5

Redes de Gas

Programación Avanzada

Formulación del modelo

El método para el diseño y análisis de redes de gas natural en régimen permanente es el propuesto por M. A. Stoner, que debe de satisfacer la ley de la conservación de la masa para cada uno de sus nodos.

$$F_I = \sum_{j\,/\,(i,j)\in ep} s_{ij}\, q_{ij} + Q_I$$

$$q_{ij} = C_{ij} \left| p_i^2 - p_j^2 \right|^n$$

Aplicando la ecuación de la ley de conservación de la masa en la red ejemplo tenemos:

$$F_1 = S_{12} C_{12} \left| p_1^2 - p_2^2 \right|^n + Q_1 = 0$$

$$F_2 = S_{21} C_{21} \left| p_2^2 - p_1^2 \right|^n + S_{23} C_{23} \left| p_2^2 - p_3^2 \right|^n + S_{24} C_{24} \left| p_2^2 - p_4^2 \right|^n = 0$$

$$F_3 = S_{32} C_{32} \left| p_3^2 - p_2^2 \right|^n + S_{34} C_{34} \left| p_3^2 - p_4^2 \right|^n + S_{35} C_{35} \left| p_3^2 - p_5^2 \right|^n = 0$$

$$F_4 = S_{42} C_{42} \left| p_4^2 - p_2^2 \right|^n + S_{43} C_{43} \left| p_4^2 - p_3^2 \right|^n + S_{46} C_{46} \left| p_4^2 - p_6^2 \right|^n = 0$$

$$F_5 = S_{53} C_{53} \left| p_5^2 - p_3^2 \right|^n + Q_5 = 0$$

$$F_6 = S_{64} C_{64} \left| p_6^2 - p_4^2 \right|^n + Q_6 = 0$$

Ariel Guzman 2001

6

Redes de Gas

Programación Avanzada

Sustituyendo los valores de la red ejemplo, en la expresiones anteriores tenemos:

$$Q_{ij} = 0.84269 E \left[\frac{P_i^2 - P_j^2}{0.6215 L} \right]^{0.5} d_{ij}^{2.667} \qquad C_{ij} = \frac{0.84269 E d_{ij}^{2.667}}{(0.6215 L_{ij})^{0.5}}$$

$$C_{12} = \frac{0.84269 E d_{12}^{2.667}}{(0.6215 L_{12})^{0.5}} = \frac{(0.84269)(0.8)(10)^{2.667}}{(0.6215 \times 30)^{0.5}} = 72.47$$

$$C_{23} = 56.52; \quad C_{23} = 23.96; \quad C_{34} = 41.50;$$

$$C_{35} = 15.42; \quad C_{46} = 19.91$$

Ariel Guzman 2001

7

Redes de Gas

Programación Avanzada

Finalmente se llega al sistema de ecuaciones siguiente:

$$F_1 = 18000 - 72.47\,\Delta p_{12} = 0$$

$$F_2 = 72.47\,\Delta p_{21} - 56.52\,\Delta p_{23} - 23.96\,\Delta p_{24} = 0$$

$$F_3 = 56.52\,\Delta p_{32} - 41.50\,\Delta p_{34} - 15.42\,\Delta p_{35} = 0$$

$$F_4 = 23.96\,\Delta p_{42} + 41.50\,\Delta p_{43} - 19.91\,\Delta p_{64} = 0$$

$$F_5 = 15.42\,\Delta p_{63} + Q_5 = 0$$

$$F_6 = 19.91\,\Delta p_{64} + Q_6 = 0$$

Ariel Guzman 2001

8

Redes de Gas

Programación Avanzada

Sistema de ecuaciones a resolver

$$\sum_{j=1}^{N}\left(\frac{\partial F_i}{\partial X_j^k}\,\Delta x_j^{k+1}\right) = -F_i\left(x_1^k, x_2^k, \ldots\ldots, x_n^k\right)$$

$$\begin{bmatrix} \frac{\partial F_1}{\partial x_1} & \frac{\partial F_1}{\partial x_2} & \ldots\ldots & \frac{\partial F_1}{\partial x_n} \\ \frac{\partial F_2}{\partial x_1} & \frac{\partial F_2}{\partial x_2} & \ldots\ldots & \frac{\partial F_2}{\partial x_n} \\ \vdots & \vdots & \vdots\vdots\vdots & \vdots \\ \frac{\partial F_n}{\partial x_1} & \frac{\partial F_n}{\partial x_2} & \ldots & \frac{\partial F_n}{\partial x_n} \end{bmatrix} \begin{bmatrix} \Delta x_1^{k+1} \\ \Delta x_2^{k+1} \\ \vdots \\ \Delta x_n^{k+1} \end{bmatrix} = \begin{bmatrix} -F_1 \\ -F_2 \\ \vdots \\ -F_n \end{bmatrix}$$

Ariel Guzman 2001

9

El cual se puede resolver fácilmente usando el método de eliminación Gaussiana.

Redes de Gas

Programación Avanzada

Las derivadas perciales cuando los elementos son conectores, están dadas por:

$$\frac{\partial F_i}{\partial p_i} = -\sum_{j/(ij)\in p}\frac{\partial F_j}{\partial p_i} \quad , \quad \frac{\partial F_i}{\partial p_j} = -2nC_{ij}\,p_j\left|p_i^2 - p_j^2\right|^{n-1}$$

$$\frac{\partial F_i}{\partial Q_i} = 1 \quad , \quad \frac{\partial F_i}{\partial Q_j} = 0$$

Ariel Guzman 2001

10

Redes de Gas

Programación Avanzada

$$\frac{72.47\,p_1}{\Delta p_{12}}\Delta p_1 - \frac{72.47\,p_2}{\Delta p_{12}}\Delta p_2 = -F_1$$

$$-\frac{72.47\,p_1}{\Delta p_{21}}\Delta p_1 - \sum_2 \Delta p_2 - \frac{56.52\,p_3}{\Delta p_{23}}\Delta p_3 - \frac{23.96\,p_4}{\Delta p_{24}}\Delta p_4 = -F_2$$

$$-\frac{56.52\,p_2}{\Delta p_{32}}\Delta p_2 - \sum_3 \Delta p_3 - \frac{41.50\,p_4}{\Delta p_{34}}\Delta p_4 = -F_3$$

$$-\frac{23.96\,p_2}{\Delta p_{42}}\Delta p_2 - \frac{41.50\,p_3}{\Delta p_{43}}\Delta p_3 - \sum_4 \Delta p_4 = -F_4$$

$$-\frac{15.42\,p_3}{\Delta p_{53}}\Delta p_3 + \Delta Q_5 = -F_5$$

$$-\frac{19.91\,p_4}{\Delta p_{64}}\Delta p_4 + \Delta Q_6 = -F_6$$

Ariel Guzmán 2001

11

Redes de Gas

Programación Avanzada

$$\begin{aligned}
311.73\Delta p_1 - 320.05\Delta p_2 &= -53644\\
-311.73\Delta p_1 + 571.83\Delta p_2 - 208.34\Delta p_3 - 56.57\Delta p_4 &= 82563\\
-200.53\Delta p_2 + 366.88\Delta p_3 - 122.81\Delta p_4 &= 40828\\
-51.25\Delta p_2 - 115.59\Delta p_3 + 236.60\Delta p_4 &= -146561\\
-42.95\Delta p_3 + \Delta Q_5 &= 85709\\
-57.22\Delta p_4 + \Delta Q_6 &= -889.46
\end{aligned}$$

$$\Delta p_1 = -70; \quad \Delta p_2 = -50; \quad \Delta p_3 = -50; \quad \Delta p_4 = -98; \quad \Delta Q_5 = 6450; \quad Q_6 = -6450$$

$$p_1 = 750 + (-70)(0.5) = 751;$$
$$p_2 = 745; \quad p_3 = 775; \quad p_4 = 801;$$
$$Q_5 = -9775; \quad Q_6 = -8225.$$

Ariel Guzmán 2001

12

Redes de Gas

Programación Avanzada

Valores iniciales supuestos:

$p_1 = 750;$ $p_2 = 770;$ $p_3 = 800;$ $p_4 = 850;$ $Q_5 = -13000,$ $Q_6 = -5000$

Sustituyendo valores, tenemos:

$$F_1 = 18000 - 72.47(174.356) = 5364.4$$
$$F_2 = 72.47(174.356) - 56.52(217.025) - 23.96(360) = -8256.27$$
$$F_3 = 56.52(217.025) - 41.50(287.228) - 15.42(287.228) = -4082.765$$
$$F_4 = 23.96(360) + 41.50(287.228) - 19.91(295.804) = 14656.10$$
$$F_5 = 15.42(287.228) + (-13000) = -8570.94$$
$$F_6 = 19.91(295.804) + (-5000) = 889.458$$

Ariel Guzman 2001

13

Redes de Gas

Programación Avanzada

Resultados obtenidos durante el Proceso Iterativo

INCÓGNITAS	ITERACIÓN 1			ITERACIÓN 2			ITERACIÓN 3		
	x_i	F_i	Δx_i	x_i	F_i	Δx_i	x_i	F_i	Δx_i
P_1	750	5364.4	-70	715	2833.2	-67.5	681.25	1543.4	-46.36
P_2	770	-8256.3	-50	745	-3952.5	-53.8	718.1	-2081.95	-37.25
P_3	800	-4082.8	-50	775	-1714.5	-51.0	749.5	-661.9	-35.56
P_4	850	14656.1	-98	801	7280.3	-69.5	766.25	3618.9	-43.11
Q_5	-13000	-8570.9	6450	-9775	-4391.8	2646	-8452	-2269.6	1244
Q_6	-5000	889.5	-6450	-8225	-54.7	-2646	-9548	-148.9	-1244

INCÓGNITAS	ITERACIÓN 4			ITERACIÓN 5					
	x_i	F_i	Δx_i	x_i	F_i	Δx_i	x_i	F_i	Δx_i
P_1	634.89	178.95	6.084	628.806	3.068	-0.069	628.737	-0.1276	0.00027
P_2	680.85	-159.68	4.782	676.068	-2.628	-0.048	676.020	0.1633	-0.00040
P_3	713.94	257.50	4.627	709.313	3.822	-0.047	709.266	-0.1263	-0.00015
P_4	723.14	-57.42	4.053	719.087	-2.196	-0.036	719.051	0.1096	-0.00049
Q_5	-7708	-94.89	-15.5	-7223.5	-1.159	0.058	-7223.44	-0.0047	-0.0012
Q_6	-10792	124.47	15.5	-10776.5	-0.907	-0.058	-10776.56	-0.001167	-0.0012

Ariel Guzman 2001

14

Redes de Gas

Programación Avanzada

Resultados finales después de la sexta iteración

INCÓGNITAS	VALORES FINALES	
	x_i	F_i
P_1	628.737	0.010
P_2	676.020	-0.021
P_3	709.266	0.026
P_4	719.051	-0.015
Q_5	-7223.44	-0.0004
Q_6	-10776.56	-0.0009

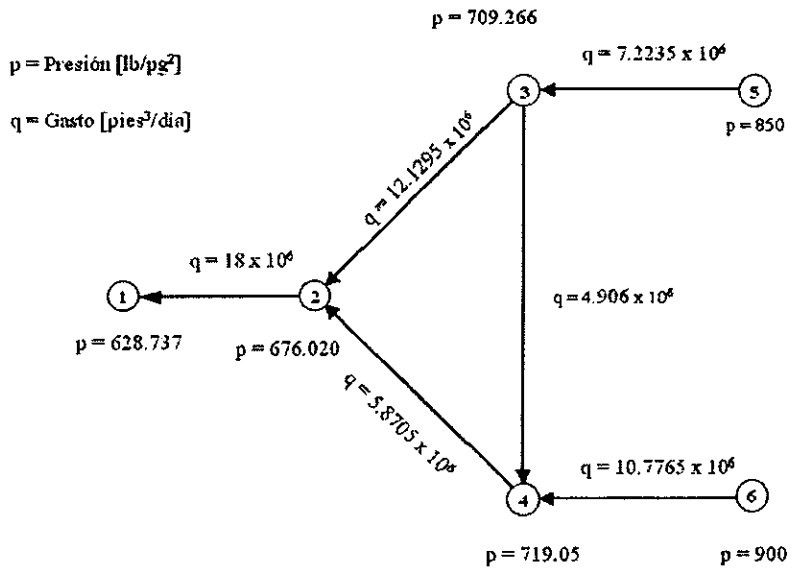
Ariel Guzman 2001

15

Redes de Gas

Programación Avanzada

Distribución final de presiones y gastos en la red de gas



Ariel Guzman 2001

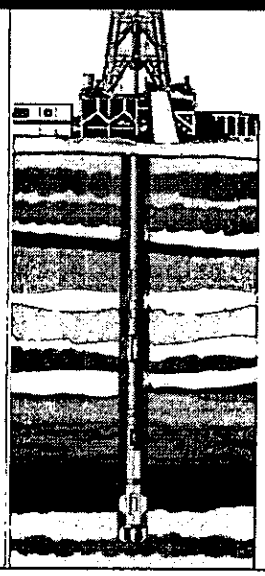
16

Universidad Nacional Autónoma de México

Facultad de Ingeniería, Ingeniería Petrolera

Programación

Avanzada



Regresión Múltiple Aplicada a la Perforación

Ariel Guzmán 2001

1

Regresión Múltiple *Aplicada a la Perforación.*

1. Introducción.

2. Modelo de Perforación y Técnica de Regresión Múltiple.

3. Optimización de la Perforación.

4. Problema.

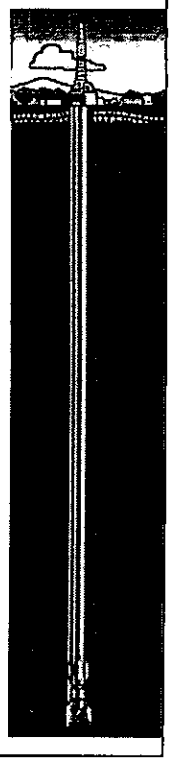
Ariel Guzmán 2001

2

Introducción.

El modelo matemático analiza ciertos parámetros que intervienen en la perforación, determinando los coeficientes de regresión a_j y así aplicar las ecuaciones para la optimización de la perforación.

Este modelo nos ayuda a calcular la selección del peso sobre barrena, velocidad de rotación y la hidráulica.



Abril Guzmán 2009

3

Modelo de Perforación.

- Velocidad de Penetración: $$\frac{dD}{dt} = Exp\left(a_1 + \sum_{j=2}^{8} a_j x_j\right)$$

- Efecto de la Resistencia de la Formación: a_1

- Efecto de la Compactación: $a_2 x_2 \;\; y \;\; a_3 x_3$

$$x_2 = 10{,}000 - D$$

$$x_3 = D^{0.69}\left(g_p - 9.0\right)$$

Abril Guzmán 2009

4

Modelo de Perforación, (continuación).

- Efecto de la Presión Diferencial:

$$a_4 x_4$$

$$x_4 = D\left(g_p - \rho_c\right)$$

- Efecto del Diámetro y Peso sobre la Barrena:

$$a_5 x_5$$

$$x_5 = \ln \frac{\frac{w}{d} - \left(\frac{w}{d}\right)_t}{4.0 - \left(\frac{w}{d}\right)_t}$$

Ariel Guzmán 2001

5

Modelo de Perforación, (continuación).

- Efecto de la Velocidad de Rotación:

$$a_6 x_6$$

$$x_6 = \ln\left(\frac{N}{100}\right)$$

- Efecto del Desgaste del Diente:

$$a_7 x_7$$

$$x_7 = -h$$

- Efecto de la Hidráulica en la Barrena:

$$a_8 x_8$$

$$x_8 = \frac{\rho q}{350\,\mu\, d_n}$$

Ariel Guzmán 2001

6

Modelo de Perforación. (continuación).

Modelo del Desgaste de la Barrena.

$$\frac{dh}{dt} = \frac{H_3}{\tau_H}\left[\frac{N}{100}\right]^{H_1}\left[\frac{\left(\frac{w}{d}\right)_{máx} - 4}{\left(\frac{w}{d}\right)_{máx} - \frac{w}{d}}\right]\left[\frac{1+\frac{H_2}{2}}{1+H_2 h}\right]$$

$$\frac{dB}{dt} = \frac{1}{\tau_B}\left[\frac{N}{100}\right]\left[\frac{w}{4d}\right]^{b}$$

τ_H : Cons tan te de abrasividad [hrs]

τ_B : Cons tan te del cojinete [hrs]

Arel Guzmán 2009

7

Modelo de Perforación. (continuación).

Las ecuaciones anteriores definen una relación entre la velocidad de penetración y las variables de perforación discutidas, pero las constantes a_2 hasta la a_8 deben ser determinadas antes de que estas ecuaciones puedan ser aplicadas. Estas constantes son determinadas por medio del análisis de regresión multiple.

Arel Guzmán 2009

8

Optimización de la Perforación.

$$Cf = \frac{Cb + Cr(Tt + Tc + Tb)}{\Delta D}$$ Costo de perforación por pie perforado (Cf) [$/pie]:

- Peso Sobre Barrena y Velecidad de Rotación:

$$Cf = \frac{Cb + Cr(Tt + Tc + Tb)}{\left[\frac{J_1 J_2 H_2}{a_7^2}\left[1 + \frac{a_7}{H_2} + (U-1)Exp\left(\frac{a_7}{H_2} + U\right)\right]\right]}$$

$$J_1 = Exp\left(a_1 + \sum_{J=2}^{6} a_J x_J + a_8 x_8\right)$$ (J) Función del peso sobre barrena por pulgada y la velocidad de rotación:

Ariel Guzmán 2001

9

Optimización de la Perforación, (continuación).

$$J_2 = \frac{\tau_H}{H_3}\left[\frac{\left(\frac{w}{d}\right)_{máx} - \frac{w}{d}}{\left(\frac{w}{d}\right)_{máx} - 4}\right]\left[\frac{100}{N}\right]^{H_1}\left[\frac{1}{1 + \frac{H_2}{2}}\right]$$

$$U = -\frac{a_7}{H_2}\sqrt{1 + 2H_2\left(\frac{Tb}{J_2}\right)}$$ (U) Función del peso sobre barrena por pulgada, la velocidad de rotación y del tiempo de rotación.

$$Tb = J_2\left[1 + \left(\frac{H_2}{2}\right)\right]$$ (Tb) Tiempo de vida de la barrena durante la rotación. [Hrs]

Ariel Guzmán 2001

10

Optimización de la Perforación, (continuación).

- Desgaste del Diente de la Barrena:

$$\left(\frac{W}{d}\right)_{opt} = \frac{a_5 H_1 \left(\frac{W}{d}\right)_{máx} + a_6 \left(\frac{W}{d}\right)_t}{a_5 H_1 + a_6}$$

Peso sobre Barrena Óptimo por pulgada.

$$Tb = \left[\frac{Cb}{Cr} + Tt + Tc\right]\left[\frac{H_1}{a_6} - 1\right]$$

Vida Esperada de la Barrena. [Hrs]

$$N_{opt} = 100\left[\frac{\tau_H \left(\frac{W}{d}\right)_{máx} - \left(\frac{W}{d}\right)_{opt}}{\tau_b H_3 \left[\left(\frac{W}{d}\right)_{máx} - 4\right]}\right]^{\frac{1}{H_1}}$$

Velocidad de Rotación Óptima. [rpm]

Ariel Guzmán 2007
11

Problema.

Cálculo de la optimización del peso sobre barrena y la velocidad de rotación.

Datos:

Tiempo de viaje = 6.0 [hrs.]
Tiempo de Conexión = 1.0 [hrs.]
Tiempo de Rotación = 120 [hrs.]
Costo de la Barrena = $400
Costo de Perforación = $500
Diámetro de la Barrena = 9.875 [pg]
Tipo de Barrena = 1-3
Peso Sobre Barrena = 1000 [lb/pg] = 4.0
Velocidad de Rotación = 100 [rpm]
Desgaste del Diente = T-6
$(W/d)_t$, 1,000 [lb/pg] = 0.5

Ariel Guzmán 2007
12

Problema, (continuación).

De un análisis de regresión se tiene que:

$a_1 = 3.5$	$a_3 = 0.002$	$a_5 = 1.2$	$a_7 = 0.9$
$a_2 = 0.0002$	$a_4 = 0.00004$	$a_6 = 0.6$	$a_8 = 0.4$

Área Química 2001
13

Problema, (continuación).

Tabla A:

Tipo de BNA	H_1	H_2	H_3	$(w/d)_{máx}$
1-1 a 1-2	1.90	7	1.00	7.0
1-3 a 1-4	1.84	6	0.80	8.0
2-1 a 2-2	1.80	5	0.60	8.5
2-3	1.76	4	0.48	9.0
3-1	1.70	3	0.36	10.0
3-2	1.65	2	0.26	10.0
3-3	1.60	2	0.20	10.0
4-1	1.50	2	0.18	10.0
Insertos	1.50	1	0.02	Ver tabla B

Área Química 2001
14

Problema, (continuación).

Tabla B: Peso de diseño maximo sobre barrena, 1,000 [lb/pg]

	Tipo de Barrena – Subtipo								Barrenas de Insertos				
Tamaño de la Barrena	1–1	1–2	1–3	1–4	2–1 2–2	2–3	3	4	5	6	7	8	9
6 1/8		5.6	6.0	6.6	6.9		7.9						
6 ¾		5.7	6.1	6.6	7.1	7.2	8.5		3.1	4.4	4.5	5.2	4.0
7 7/8	6.0	6.2	6.6	7.0	7.5	7.6	8.7	9.4	3.5	4.5	5.0	5.7	4.6
8 ¾	6.2	6.5	6.8	7.2	7.8	8.0	9.5	10.0	3.7	5.1	5.2	5.8	4.7
9 7/8	6.5	6.7	7.1	7.0	7.6	7.7	8.9		3.6	5.1	5.1	5.9	4.6
10 3/8		6.4		7.0			8.8		3.5	5.0	5.0	5.8	4.6
12 ¼	5.9	6.1	6.4	6.7	7.3	7.4	8.5		3.5	4.9	4.9	5.6	4.4
14 ¾ - 15		5.3		5.8		6.3	7.4		3.4	4.7	4.8	5.4	4.3
17 ½		5.0		5.7			7.0		3.0	4.2	4.2	4.8	3.8

Angel Guzmán 2009

15

Problema, (continuación).

Solución:

1. *Se calcula la constante de abrasividad de la formación de la ecuación dh/dt y se obtienen los valores de H de la tabla A:*

   $H_1 = 1.84$ $H_2 = 6$ $H_3 = 0.8$

   $(w/d)_{máx} = 8.0$ $\tau_H = 15.7$ [hrs]

2. *Calcular el peso óptimo sobre barrena:*

   (w/d) = 6400 [lb/pg]

3. *Calcular la vida esperada de la barrena:*

   Tb = 16.1 [hrs]

4. *Cálculo de la Velocidad de Rotación Óptima:*

   N_{opt} = 60 [rpm]

Angel Guzmán 2009

16

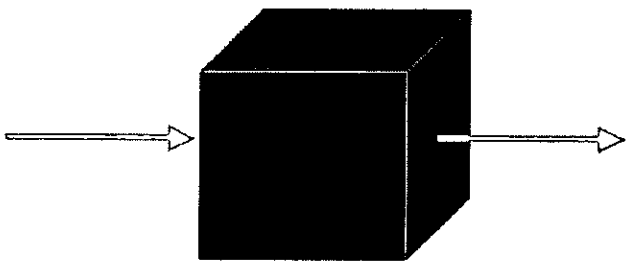
Programación Avanzada

Simulador de un yacimiento de gas en coordenadas X,Y.

Universidad Nacional Autónoma de México

Facultad de Ingeniería, Ingeniería Petrolera

Ecuación de Conservación de Masa para un Intervalo de Tiempo.



$m^{n+1} = m^n$ + (m entra en el intervalo de tiempo) – (m sale en el intervalo de tiempo)

donde:

$$m = (\Delta x \Delta y h \phi)\rho$$

Ariel Guzmán 2001

2

Ecuación de Conservación de Masa para un Intervalo de Tiempo.

La ley de Darcy para el flujo de masa* dentro de una celda es (masa que entra a la celda durante el intervalo de tiempo):

$$m^{entra} = \left(\frac{\rho}{\mu}\frac{0.00633\ kA}{\Delta x}\right)\left(p_{i+1}^{n+1} - p_i^{n+1}\right)\Delta t + \left(\frac{\rho}{\mu}\frac{0.00633\ kA}{\Delta x}\right)\left(p_{i-1}^{n+1} - p_i^{n+1}\right)\Delta t$$

donde:

A = Área de la sección transversal de flujo

La masa que sale de la celda durante el intervalo de tiempo es:

$$m^{sale} = \rho_{sc}\, q\, \Delta t$$

donde:

q = gasto producido (signo positivo para producción, signo negativo para inyección)

Ariel Guzman 2001

3

* **Nota:** Se ignoran los efectos de gravedad para simplificar.

Ecuación de Conservación de Masa para un Intervalo de Tiempo.

Si reorganizamos la ecuación y dividimos entre Δt obtenemos la siguiente ecuación:

$$\left(\frac{\rho}{\mu}\frac{0.00633\ k\Delta yh}{\Delta x}\right)\left(p_{i-1}^{n+1} - p_i^{n+1}\right) + \left(\frac{\rho}{\mu}\frac{0.00633\ k\Delta yh}{\Delta x}\right)\left(p_{i+1}^{n+1} - p_i^{n+1}\right) = \left(\frac{\Delta x\Delta yh}{\Delta t}\right)\left[(\rho\phi)_i^{n+1} - (\rho\phi)_i^{n}\right] + \rho_{sc} q$$

si reducimos la ecuación:

$$\left(\frac{\rho}{\mu}T_W\right)\left(p_{i-1}^{n+1} - p_i^{n+1}\right) + \left(\frac{\rho}{\mu}T_E\right)\left(p_{i+1}^{n+1} - p_i^{n+1}\right) = \left(\frac{\Delta x\Delta yh}{\Delta t}\right)\left[(\rho\phi)_i^{n+1} - (\rho\phi)_i^{n}\right] + \rho_{sc} q$$

donde:

$$T_W = T_E = \frac{0.00633\ k\Delta yh}{\Delta x}$$

Ariel Guzman 2001

4

Los valores de densidad y viscosidad se calculan al tiempo n.

Ecuación de Conservación de Masa para un Intervalo de Tiempo.

El término del lado derecho de la ecuación lo ponemos en función de la presión:

$$\left(\frac{\rho}{\mu}T_W\right)\left(p_{i-1}^{n+1}-p_i^{n+1}\right)+\left(\frac{\rho}{\mu}T_E\right)\left(p_{i+1}^{n+1}-p_i^{n+1}\right)=\left(\frac{V_p^n\rho^n c_t}{\Delta t}\right)\left(p_i^{n+1}-p_i^n\right)+\rho_{sc}q$$

donde:

$$V_p=\Delta x\Delta y h\,\phi^n$$

$$c_t=\frac{1}{\rho^n\phi^n}\frac{(\rho\phi)^{n+1}-(\rho\phi)^n}{\left(p^{n+1}-p^n\right)}$$

Esta ecuación tiene la forma similar de la ecuación de difusividad, excepto por el término de producción.

Ecuación de Conservación de Masa para un Intervalo de Tiempo.

La forma de la matriz que se obtiene de la ecuación en diferencias finitas forma un sistema tridiagonal para 1D.

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_4 & b_4 & c_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & \bullet & \bullet & \bullet & 0 & 0 \\ 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & 0 \\ 0 & 0 & 0 & 0 & 0 & a_{m-1} & b_{m-1} & c_{m-1} \\ 0 & 0 & 0 & 0 & 0 & 0 & a_m & b_m \end{bmatrix}\begin{bmatrix} p_1^{n+1} \\ p_2^{n+1} \\ p_3^{n+1} \\ p_4^{n+1} \\ \bullet \\ \bullet \\ p_{m-1}^{n+1} \\ p_m^{n+1} \end{bmatrix}=\begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ \bullet \\ \bullet \\ d_{m-1} \\ d_m \end{bmatrix}$$

Simulador de un Yacimiento de Gas en coordenadas X,Y.

La ecuación en diferencias finitas para el flujo de un gas en términos de la pseudo-presión del gas, $p_p(p)$ se forma de la siguiente manera:

Tenemos:

$$\left(\frac{\rho}{\mu}T_w\right)\left(p_{i-1}^{n+1}-p_i^{n+1}\right)+\left(\frac{\rho}{\mu}T_E\right)\left(p_{i+1}^{n+1}-p_i^{n+1}\right)=\left(\frac{\Delta x\Delta yh}{\Delta t}\right)\left[(\rho\phi)_i^{n+1}-(\rho\phi)_i^n\right]+\rho_{sc}q$$

dividimos la ecuación entre ρ_{sc}:

$$\left(\frac{T_{sc}}{P_{sc}T}\frac{p}{z\mu}T_w\right)\left(p_{i-1}^{n+1}-p_i^{n+1}\right)+\left(\frac{T_{sc}}{P_{sc}T}\frac{p}{z\mu}T_E\right)\left(p_{i+1}^{n+1}-p_i^{n+1}\right)=\frac{1}{\Delta t}\left(\frac{T_{sc}}{P_{sc}T}\right)\left[\left(\frac{V_p p}{z}\right)_i^{n+1}-\left(\frac{V_p p}{z}\right)_i^n\right]+q$$

donde:

$$\frac{\rho}{\rho_{sc}}=\frac{1}{B_g}=\frac{T_{sc}}{P_{sc}T}\frac{p}{z}$$

Ariel Guzman 2001

7

Simulador de un Yacimiento de Gas en coordenadas X,Y.

El concepto de pseudo-presión del gas, $p_p(p)$ es:

$$p_p(p)=2\int_0^p\frac{p}{z\mu}dp$$

también se denota como:

$$\Delta p_p(p)=\left(\frac{2p}{z\mu}\right)\Delta p$$

Donde la cantidad en el parentesis es la integración promedio entre el rango de presiones.

Ariel Guzman 2001

8

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Tomamos un témino de flujo de la ecuación general.

$$\left(\frac{T_{sc}}{P_{sc}T}\frac{p}{z\mu}T_E\right)\left(p_{i+1}^{n+1}-p_i^{n+1}\right)$$

reacomodamos los términos:

$$\left(\frac{T_{sc}}{P_{sc}T}\frac{1}{2}T_E\right)\left(\frac{2p}{z\mu}\right)\left(p_{i+1}^{n+1}-p_i^{n+1}\right)$$

y finalmente tenemos:

$$\left(\frac{T_{sc}}{P_{sc}T}\frac{1}{2}T_E\right)\left(p_{p_{i+1}}^{n+1}-p_{p_i}^{n+1}\right)$$

Ariel Guzman 2001

9

Simulador de un Yacimiento de Gas en coordenadas X,Y.

SÍ definimos un coefiente de flujo de la ecuación anterior

$$a_E=\frac{T_{sc}}{P_{sc}T}\frac{1}{2}T_E$$

el témino de flujo queda como:

$$a_E\left(p_{p_{i+1}}^{n+1}-p_{p_i}^{n+1}\right)$$

la ecuación general en términos de la pseudo-presión del gas es:

$$a_E\left(p_{p_{i+1}}^{n+1}-p_{p_i}^{n+1}\right)+a_W\left(p_{p_{i-1}}^{n+1}-p_{p_i}^{n+1}\right)=\frac{1}{\Delta t}\left(\frac{T_{sc}}{P_{sc}T}\right)\left[\left(\frac{V_p p}{z}\right)_i^{n+1}-\left(\frac{V_p p}{z}\right)_i^{n}\right]+q$$

Ariel Guzman 2001

10

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Si simplificamos el lado derecho de la ecuación:

$$\frac{1}{\Delta t}\left(\frac{T_{sc}}{P_{sc}T}\right)\left[\left(\frac{V_p p}{z}\right)_i^{n+1}-\left(\frac{V_p p}{z}\right)_i^{n}\right]=\alpha\left(p_{p_i}^{n+1}-p_{p_i}^{n}\right)$$

donde:

$$\alpha=\frac{1}{\Delta t}\left(\frac{T_{sc}}{P_{sc}T}\right)\left[\frac{\left(\frac{V_p p}{z}\right)_i^{n+1}-\left(\frac{V_p p}{z}\right)_i^{n}}{\left(p_{P_i}^{n+1}-p_{P_i}^{n}\right)}\right]$$

finalmente tenemos:

$$a_E\left(p_{p_{i+1}}^{n+1}-p_{p_i}^{n+1}\right)+a_W\left(p_{p_{i-1}}^{n+1}-p_{p_i}^{n+1}\right)=\alpha\left(p_{p_i}^{n+1}-p_{p_i}^{n}\right)+q$$

Ariel Guzmán 2001

11

Simulador de un Yacimiento de Gas en coordenadas X,Y.

La ecuación en diferencias finitas del gas la extendemos a 2D:

$$a_E\left(p_{p_{i+1,j}}^{n+1}-p_{p_{i,j}}^{n+1}\right)+a_S\left(p_{p_{i,j+1}}^{n+1}-p_{p_{i,j}}^{n+1}\right)+$$

$$a_W\left(p_{p_{i-1,j}}^{n+1}-p_{p_{i,j}}^{n+1}\right)+a_N\left(p_{p_{i,j-1}}^{n+1}-p_{p_{i,j}}^{n+1}\right)=$$

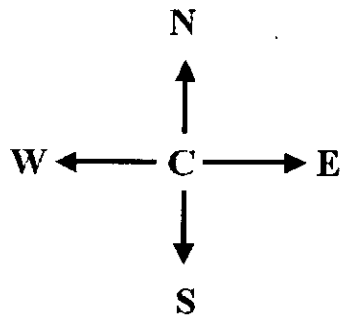
$$\alpha\left(p_{p_{i,j}}^{n+1}-p_{p_{i,j}}^{n}\right)+q$$

Ariel Guzmán 2001

12

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Las direcciones presentadas en las ecuaciones anteriores se definen en el siguiente diagrama:



en la ecuación nótese que los coeficientes son simétricos:

$$a_{E_{i,j}} = a_{W_{i+1,j}} \qquad a_{S_{i,j}} = a_{N_{i,j+1}}$$

Ariel Guzman 2001

13

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Si rearreglamos la ecuación y dejamos las incognitas en el lado izquierdo:

$$-a_N p^{n+1}_{p_{i,j-1}} - a_W p^{n+1}_{p_{i-1,j}} - a_C p^{n+1}_{p_{i,j}} - a_E p^{n+1}_{p_{i+1,j}} - a_S p^{n+1}_{p_{i,j+1}} = d$$

donde:

$$a_C = a_N + a_W + a_E + a_S + \alpha$$

$$d = \alpha\, p^{n}_{p_{i,j}} - q$$

Ariel Guzman 2001

14

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Condiciones Iniciales.

Las condiciones iniciales son implementadas asignando valores especificos de presión y su correspondiente $p_p(p)$ para cada celda. Estos valores usualmente son iguales para todas las celdas.

Condiciones de Frontera.

Las condiciones de frontera es una condición donde se establece que no pasa flujo a través de esta. Estas se implementan simplemente asignando a los coeficientes igual a cero. Por ejemplo, $a_E = 0$ en la frontera en la dirección indicada.

Heterogenidad, Anisotropía y no uniformidad de las celdas.

Para el caso general cuando los valores de Δx, Δy, k, h y ϕ varian con respecto a su localización los coeficientes de flujo son redefinidos. También la permeabilidad puede ser anisotrópica (dirección).

Ariel Guzman 2001

15

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Los coeficientes podemos definirlos como (propiedades promedio armónicas):

$$T_{E_{i,j}} = \frac{T_1 T_2}{T_1 + T_2}$$

donde:

$$T_1 = \frac{0.00633 \quad \Delta y_j (k_x h)_{i,j}}{\Delta x_i / 2}$$

$$T_2 = \frac{0.00633 \quad \Delta y_j (k_x h)_{i+1,j}}{\Delta x_{i+1} / 2}$$

Ariel Guzman 2001

16

Simulador de un Yacimiento de Gas en coordenadas X,Y.

También:

$$T_{S_{i,j}} = \frac{T_3 T_4}{T_3 + T_4}$$

donde:

$$T_3 = \frac{0.00633 \quad \Delta x_i \left(k_y h\right)_{i,j}}{\Delta y_i / 2}$$

$$T_2 = \frac{0.00633 \quad \Delta x_i \left(k_y h\right)_{i,j+1}}{\Delta y_{j+1} / 2}$$

la simetría de estos coeficientes es:

$$T_{W_{i,j}} = T_{E_{i-1,j}} \qquad T_{N_{i,j}} = T_{S_{i,j-1}}$$

Ariel Guzmán 2001

17

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Efectos Gravitacionales.

Si los efectos gravitacionales son incluidos en el flujo de gas, la aplicación de la ley de Darcy debe ser modificada. Como se a manejado hasta el momento un término de flujo de la ecuación es:

$$\left(\frac{\rho}{\mu} T_E\right)\left(p_{i+1}^{n+1} - p_i^{n+1}\right)$$

Este término solo incluye el flujo horizontal. Cuando los efectos gravitacionales son incluidos el término cambia de la siguiente manera:

$$\left(\frac{\rho}{\mu} T_E\right)\left(p_{i+1}^{n+1} - p_i^{n+1}\right) + \left(\frac{\rho^2}{144\mu} T_E\right)\left(Z_{i+1} - Z_i\right)$$

Ariel Guzmán 2001

18

Donde Z es igual a la elevación en pies, y es positivo en la dirección hacia arriba.

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Si lo anterior lo extendemos a las cuatro direcciones, el trémino por efectos de gravitacionales, G, es el siguiente:

$$G=\left(\frac{\rho^2}{144\mu}T_E\right)\left(Z_{i+1,j}-Z_{i,j}\right)+\left(\frac{\rho^2}{144\mu}T_W\right)\left(Z_{i-1,j}-Z_{i,j}\right)+\left(\frac{\rho^2}{144\mu}T_S\right)\left(Z_{i,j+1}-Z_{i,j}\right)+\left(\frac{\rho^2}{144\mu}T_N\right)\left(Z_{i,j-1}-Z_{i,j}\right)$$

Donde ρ y μ son promedio en la dirección indicada. Finalmente el término de la ecuación general " *d* " queda de la siguiente manera:

$$d = \alpha p^{n}_{p_{i,j}} - q - \frac{G}{\rho_{sc}}$$

Normalmente el término por efecto gravitacional se considera despreciable en la ingeniería de yacimientos de gas porque las densidades del gas son muy bajas.

Ariel Guzman 2001

19

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Presión de Fondo Fluyendo.

Las ecuaciones antes mencionadas describen la presión, $p_p(p)$, en el centro de las celda, esta presión respresenta el promedio en toda la celda. Sí los pozos se localizan terminados en el centro de la celda, la $p_{i,j}$ no es la presión de fondo fluyendo. Las anteriores ecuaciones calculan el flujo de gas de celda a celda, pero no representa los gradientes de presión dentro del pozo. La ecuación para calcular la presión de fondo fluyendo, P_{wf} es:

$$q = J'\left(p^{n+1}_{p_{i,j}} - p_{p_{wf}}\right)$$

donde:

$$J' = \frac{0.01988\, khT_{sc}}{p_{sc}T\left(\ln\frac{r_o}{r_w}+s+0.75\right)} \qquad k = \sqrt{k_x k_y}$$

Para un medio isotrópico

$$r_o = \frac{0.28\left(\sqrt{\frac{k_y}{k_x}}\Delta x^2 + \sqrt{\frac{k_x}{k_y}}\Delta y^2\right)^{1/2}}{\left(\frac{k_y}{k_x}\right)^{1/4}+\left(\frac{k_x}{k_y}\right)^{1/4}} \qquad r_o = 0.14\sqrt{\Delta x^2+\Delta y^2}$$

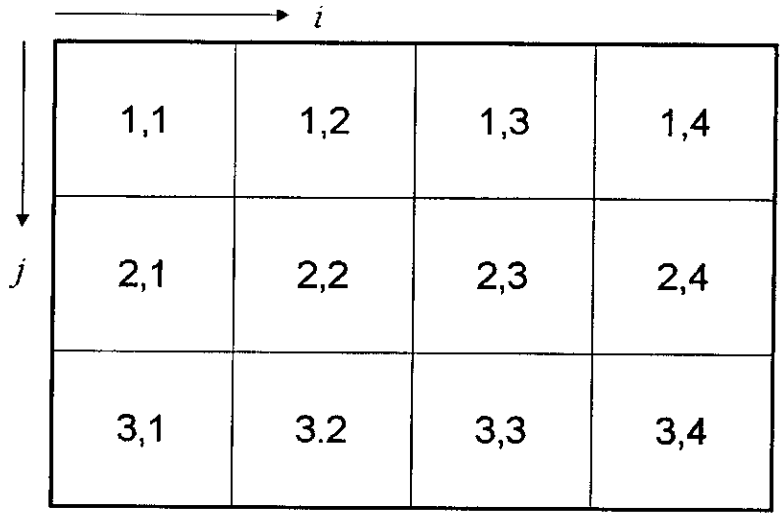
Ariel Guzman 2001

20

Simulador de un Yacimiento de Gas en coordenadas X,Y.

Sistema de Ecuaciones.

Si tomamos la siguiente malla como ejemplo el sistema de ecuaciones que se forma es:



Ariel Guzmán 2001

21

Simulador de un Gas Real en coordenadas X,Y.

$$\begin{bmatrix} a_{C_1} & -a_{E_1} & 0 & 0 & -a_{S_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -a_{W_2} & a_{C_2} & -a_{E_2} & 0 & 0 & -a_{S_2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -a_{W_3} & a_{C_3} & -a_{E_3} & 0 & 0 & -a_{S_3} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -a_{W_4} & a_{C_4} & 0 & 0 & 0 & -a_{S_4} & 0 & 0 & 0 & 0 \\ -a_{N_5} & 0 & 0 & 0 & a_{C_5} & -a_{E_5} & 0 & 0 & -a_{S_5} & 0 & 0 & 0 \\ 0 & -a_{N_6} & 0 & 0 & -a_{W_6} & a_{C_6} & -a_{E_6} & 0 & 0 & -a_{S_6} & 0 & 0 \\ 0 & 0 & -a_{N_7} & 0 & 0 & -a_{W_7} & a_{C_7} & -a_{E_7} & 0 & 0 & -a_{S_7} & 0 \\ 0 & 0 & 0 & -a_{N_8} & 0 & 0 & -a_{W_8} & a_{C_8} & 0 & 0 & 0 & -a_{S_8} \\ 0 & 0 & 0 & 0 & -a_{N_9} & 0 & 0 & 0 & a_{C_9} & -a_{E_9} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -a_{N_{10}} & 0 & 0 & -a_{W_{10}} & a_{C_{10}} & -a_{E_{10}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -a_{N_{11}} & 0 & 0 & -a_{W_{11}} & a_{C_{11}} & -a_{E_{12}} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -a_{N_{12}} & 0 & 0 & -a_{W_{12}} & a_{C_{12}} \end{bmatrix} \begin{bmatrix} p_{P_1} \\ p_{P_2} \\ p_{P_3} \\ p_{P_4} \\ p_{P_5} \\ p_{P_6} \\ p_{P_7} \\ p_{P_8} \\ p_{P_9} \\ p_{P_{10}} \\ p_{P_{11}} \\ p_{P_{12}} \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \\ d_9 \\ d_{10} \\ d_{11} \\ d_{12} \end{bmatrix}$$

Ariel Guzmán 2001

22

Simulador de un Gas Real en coordenadas X,Y.

Solución.

La ecuación en diferencias finitas del gas real es no-lineal porque el coeficiente α depende de la variable dependiente $p_p(p)$. Los coeficientes de transmisibilidad no son no-lineales, estos no cambiancon el tiempo.

La solución al procedimiento no-lineal es el siguiente:

1. Se resuelve el sistema de ecuaciones para $p_p(p)$.
2. Se reevalua α y se recalcula $a_{Cl,J}$ para cada celda.
3. Se repiten los pasos 1 y 2 hasta alcanzar una convergencia, para continuar al siguiente intervalo de tiempo.

Ariel Guzmán
2001

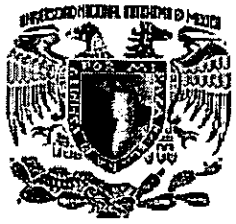
23

Universidad Nacional Autónoma de México

Facultad de Ingeniería, Ingeniería Petrolera

Programación

Avanzada



Notas sobre

Microsoft Visual Basic Versión 6.0

"Básico"

Ariel Guzmán
2001
1

¿Qué es Visual Basic?.

Es un lenguaje visualizador de eventos. Qué está basado en el lenguaje de programación Basic.

Microsoft Visual Basic Ver. 6.0

Existen tres ediciones de *Visual Basic*.

La edición de aprendizaje: Contiene todas las herramientas que se necesitan para crear una aplicación.

La edición Profesional: Tiene todas las herramientas y características de la edición de aprendizaje, además de incluir componentes completos de ActiveX®.

La edición Empresarial: Incluye todas las herramientas y características encontradas en la edición Profesional y contiene además componentes para la creación y manipulación de Bases de Datos.

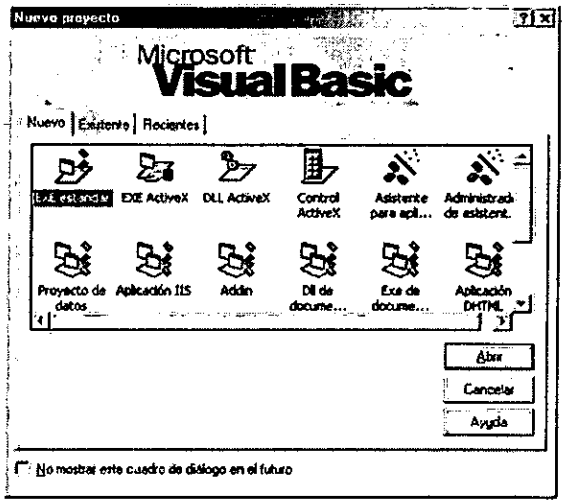
Ariel Guzmán
2001
2

Interfaz de Desarrollo.

Al iniciar *Visual Basic,* se abre una pantalla como la que se muestra a continuación.

Éste es el cuadro de diálogo Nuevo Proyecto que permite empezar a crear todos los tipos de aplicaciones diferentes de Visual Basic.
Lo que aparezca en este cuandro de dialogo, depende de la versión de Visual Basic que se tenga.

Por ejemplo: Éste cuadro de diálogo corresponde a la versión empresarial de Visual Basic.



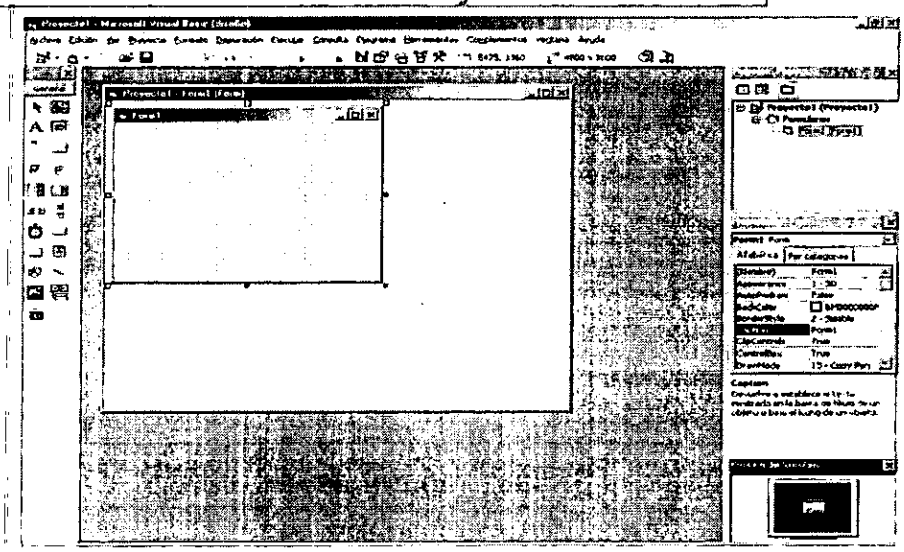
Ariel Guzmán 2001

3

Interfaz de Desarrollo.

Una vez que se ha seleccionado un tipo de proyecto (EXE estándard), aparece el ambiente gráfico de desarrollo, el cual muestra por defaul los siguientes elementos.

Microsoft Visual Basic Ver. 6.0



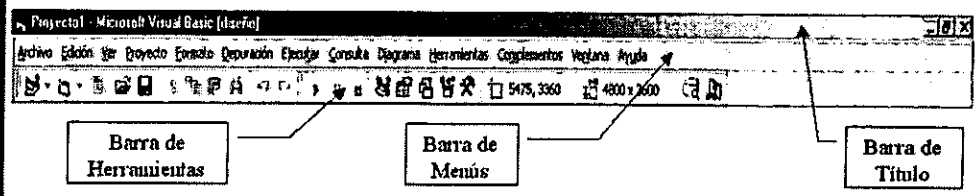
Ariel Guzmán 2001

4

Descripción de los elementos.

Barra de Herramientas, (Toolbar).

Microsoft Visual Basic Ver. 6.0



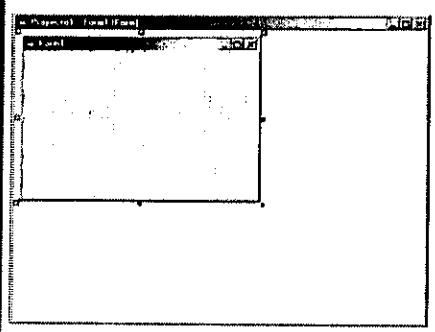
Incluye los botones de muchos de los comandos cumunes más usados por *Visual Basic,* tal como, Abrir Proyecto, Guardar Proyecto, etc. También contiene los botones que despliegan la Ventana del Explorador de Proyectos, la Ventana de Propiedades, la Caja de Herramientas entre otros elementos de *Visual Basic*.

Ariel Guzmán
2001
5

Descripción de los elementos, (continuación).

Diseñador de Formas, (Form Designer).

Microsoft Visual Basic Ver. 6.0



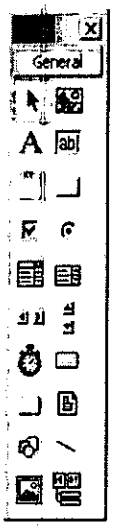
En *Visual Basic,* una forma es una ventana y es la parte principal de la aplicación basada en controles y objetos llamados *gráficos*. Es el lugar donde se comienza a construir la interfaz, es decir, donde se colocarán todos los controles y objetos que utilizará la aplicación.

Ariel Guzmán
2001
6

Descripción de los elementos, (continuación).

Caja de Herramientas (Toolbox).

Microsoft Visual Basic Ver. 6.0



Contiene todos los posibles objetos y controles que se pueden agregar a la forma para crear la interfaz de la aplicación.

Existen dos maneras de colocar los objetos y controles en la forma. La primera es por medio de un doble click en el control deseado, con esto automaticamente el control se colocará en medio de la forma y con un tamaño ya definido. La segunda es seleccionar el control que se desee utilizar, con el puntero del mouse sobre la forma y presionando el botón izquierdo dibujar el control hasta que se tenga el tamaño deseado.

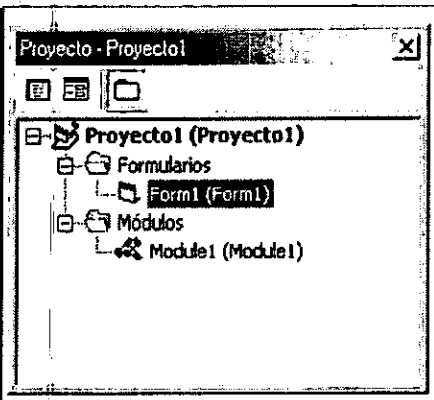
Ariel Guzmán 2001

7

Descripción de los elementos, (continuación).

Ventana del Explorador del Proyecto, (Project Explorer Window).

Microsoft Visual Basic Ver. 6.0



Contiene la lista de todos los archivos usados para construir la interfaz que conforma la aplicación. A esta lista de archivos se le llama *proyecto*. En ésta ventana se puede cambiar el modo de vista entre la forma y el código usando los botones que se encuentran en la parte superior izquierda de la ventana.

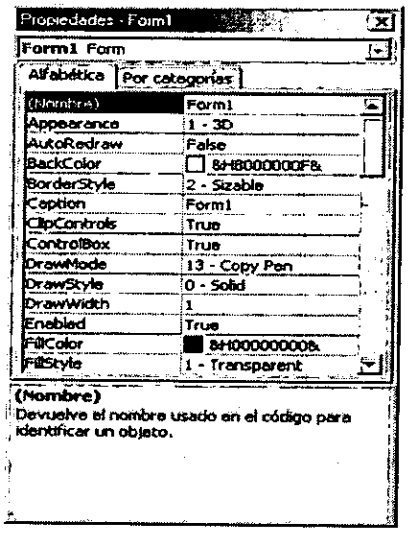
Ariel Guzmán 2001

8

Descripción de los elementos, (continuación).

Ventana de Propiedades, (Propieties Window).

Microsoft Visual Basic Ver. 6.0



Contiene la lista de todas las propiedades que se establecen para un control selecionado o forma. Una propiedad decribe una característica de un objeto, tal como, el tamaño, color, o tipo de letra, etc. Éstas se pueden modificar mientras el programa es editado, o también pueden ser modificadas en *tiempo de ejecución.*

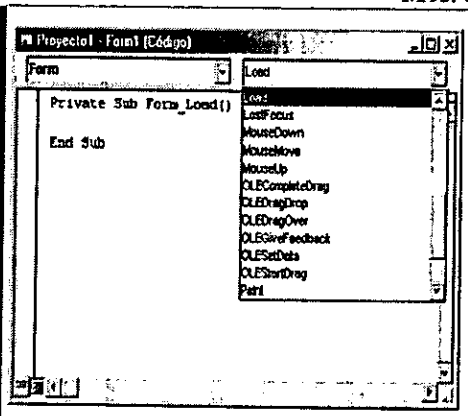
Ariel Guzmán 2001

9

Descripción de los elementos, (continuación).

Ventana de Edición del Código, (Code Editor Window).

Microsoft Visual Basic Ver. 6.0



En ésta ventana es donde se escribe el código asociado a la forma, control o a un módulo que implementa una aplicación.

Se puede asociar el código con la forma o puede estar contenida en un módulo aparte, esto último queda a criterio del programador o al tipo de aplicación que se este desarrollando, e inclusive ambos.

Ariel Guzmán 2001

10

Archivos que componen una Aplicación en Visual Basic.

Cuando se guardan todos los archivos (proyecto), *Visual Basic* genera otros que son necesarios para el funcionamiento de la aplicación.

Microsoft Visual Basic Ver. 6.0

A continuación se describen los mínimos archivos necesarios principales de una aplicación y un ejemplo:

Tipo de Archivo	Extensión	Descripción
Mínimos Archivos Necesarios que se generan para un aplicación.		
Proyecto	.vbp	Contiene los datos del diseño de la aplicación
Forma	.frm .frx	Estos archivos contienen la forma, los objetos de la forma y el código que se ejecuta cuando un evento ocurre sobre la forma.
Módulos	.bas	El o Los Módulos contienen los procedimientos Sub y Funtion que pueden ser llamados por cualquier forma u objeto sobre la forma.
Otros tipos de archivos		
Controles	.ocx	Contienen a los controles privados que forman parte de un proyecto EXE estándar
Recursos	.res	Carga datos de varios tipos posibles, mapa de bits, icono, cursor o una cadena desde un archivo de recursos (.res).
Control	.ctl	Contiene el controles ActiveX creado con Visual Basic que es definido por un módulo UserControl. El código fuente que agregue e este módulo para implementar el control ActiveX y se almacena en un archivo (.ctl.)

Ariel Guzmán 2001

11

Terminología de Visual Basic.

Como cualquier otro lenguaje de programación, *Visual Basic* requiere la comprensión de algunas terminologias comunes.

Microsoft Visual Basic Ver. 6.0

En la tabla siguiente se describen algunas de las terminologías más comunes:

Término	Descripción
Tiempo de Diseño	Es el tiempo en que una aplicación esta siendo desarrollada en el ambiente Visual Basic.
Tiempo de Ejecución	Es el tiempo en que una aplicación es ejecutada. En este tiempo el programador interactúa con la aplicación como lo haría el usuario.
Forma	Ventana que sirve como interfase para una aplicación o como caja de diálogo usada para reunir información del usuario.
Controles	Son representaciones gráficas de los objetos, tal como, botones, cajas de lista, etc., que el usuario manipula para suministrar información a la aplicación.
Objetos	Es un término general usado para describir todas las formas y controles que hacen un programa.
Propiedades	Características de un objeto, tal como, tamaño, color, etc.
Métodos	Acciones que un objeto pude desarrollar o que puede ser desarrollada sobre el objeto.
Eventos	Acciones reconocidas por una forma o control. Ocurre el evento como el usuario, sistema operativo o aplicación interactúa con el objeto de un programa.

Ariel Guzmán 2001

12

Construcción de una Aplicación en Visual Basic.

Para cear una aplicación en *Visual Basic* se sugiere la siguiente secuencia:

Microsoft Visual Basic Ver. 6.0

1) Abrir un nuevo proyecto y/o utilizar el proyecto creado cuando comienza *Visual Basic*.
2) Crear una una forma por cada ventana de la interfaz y establecer las propiedades para cada forma.
3) Crear los controles para cada forma y establecer las propiedades para cada control.
4) Escribir los procedimientos de eventos y los procedimientos generales.
5) Salvar el proyecto.
6) Probar y depurar la interfaz.
7) Hacer un archivo ejecutable (.exe).

Ariel Guzmán 2001
13

Formas.

Las formas así como los controles tienen propiedades por ser objetos.

Microsoft Visual Basic Ver. 6.0

Algunas de las propiedades más importantes son las siguientes:

Propiedad	Descripción
BorderStyler	Establece el estilo del borde.
Caption	Título de la forma.
FontSize	Tamaño del Font.
Name	Nombre usado en código. Prefijo frm
Height	Altura de la forma.
Left y Top	Determina la esquina superior izquierda de la forma
MaxButton	Habilita o Deshabilita el botón maximizar
MinButton	Habilita o Deshabilita el botón minimizar
Visible	Muestra o Esconde a la forma.
Width	Determina el ancho de la forma.
WindowState	Determina como va aparecer la forma, Maximizada, Minimizada o Normal.

Ariel Guzmán 2001
14

Métodos y Eventos de las Formas.

Como cualquier otro objeto, una forma expone métodos y responde a eventos.

Microsoft Visual Basic Ver. 6.0

Eventos:

Load.- Carga la forma en memoria. Normalmente este evento se utiliza para inicializar variables o propiedades.

Unload.- Descarga o remueve la forma de memoria. Las varaibles o propiedades también son descargadas.

Métodos:

Hide.- Remueve una forma de la pantalla, es decir, pone su propiedad *visible* en *false*. Al usar hide, solo se remueve la forma de pantalla, pero aún sigue cargada en memoria.

Show.- Despliega o muestra una forma que ha sido escondida a través del método hide.

Ariel Guzmán 2001

15

Forma MDI.

Interfaz de Documento Múltiple, (Multiple Document Interface, MDI).

Microsoft Visual Basic Ver. 6.0

La forma MDI, permite presentar varias formas o ventanas dentro de sí misma. Una forma MDI permite al usuario desplegar múltiples formas al mismo tiempo, cada forma presentada en una ventana por separado.

Ariel Guzmán 2001

16

Forma MDI, (continuación).

Interfaz de Documento Múltiple, (Multiple Document Interface, MDI).

Microsoft Visual Basic Ver. 6.0

Las características principales de la Forma *MDI* son las siguientes:

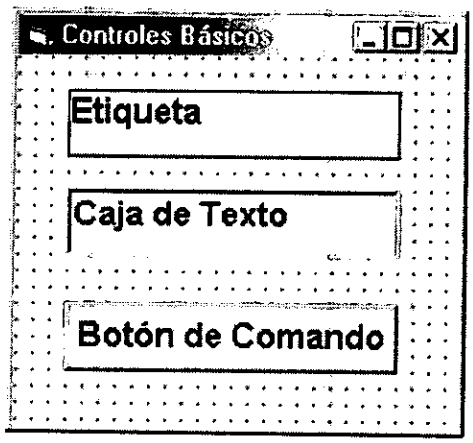
1) Existe sólo una forma padre o contenedor.
2) Las formas hijas, se crean siempre dentro del contenedor, y no pueden salir de él y se minimizan dentro del contenedor.
3) Dentro de una forma padre, no sólo pueden existir formas hijas, sino también pueden existir formas normales.

Ariel Guzmán 2001
17

Controles Básicos.

Tres de los controles comunmente más usados son:

Microsoft Visual Basic Ver. 6.0



En cualquier aplicación escrita en *Visual Basic* se usan estos tres controles.

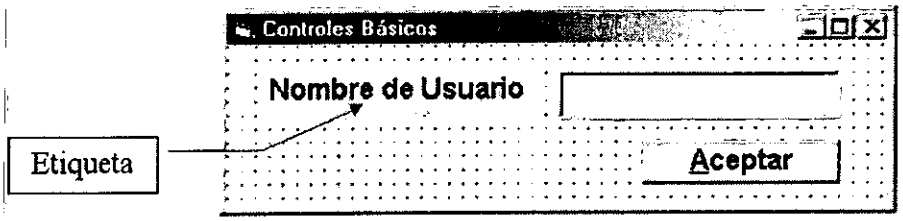
Ariel Guzmán 2001
18

Controles Básicos, (continuación).

Etiqueta, (Label).

Microsoft Visual Basic Ver. 6.0

Una etiqueta permite mostrar texto en la ventana. Una característica importante de este control es que el usuario no puede editar el texto directamente durante la ejecución. El uso más comun para este control es la de identificar el propósito de otro control.



Ariel Guzmán 2001

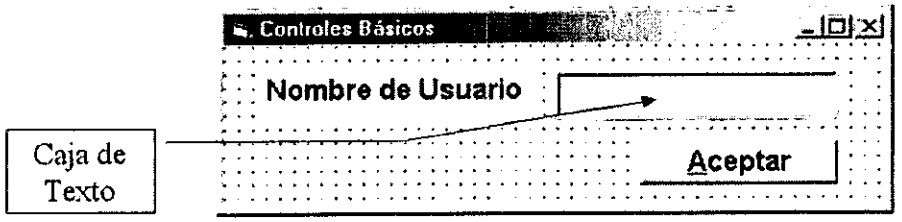
19

Controles Básicos, (continuación).

Caja de Texto, (TextBox).

Microsoft Visual Basic Ver. 6.0

Una caja de texto permite obtener información del usuaio o presentar información proveniente de la aplicación. A lo contrario de la etiqueta, el usuario puede editar el texto directamente durante la ejecución.



Ariel Guzmán 2001

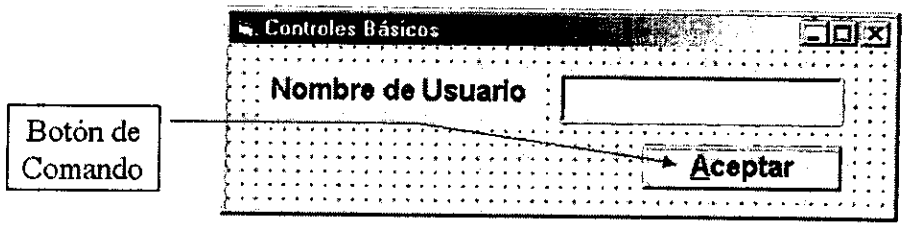
20

Controles Básicos, (continuación).

Botón de Comando, (CommandButton).

Microsoft Visual Basic Ver. 6.0

Un botón de comando desarrolla una tarea cuando el usuario presiona (click) al botón. Se usa este control para comenzar, interrumpir o para terminar un proceso. El evento mayormente usado para este control es el evento "*click*".



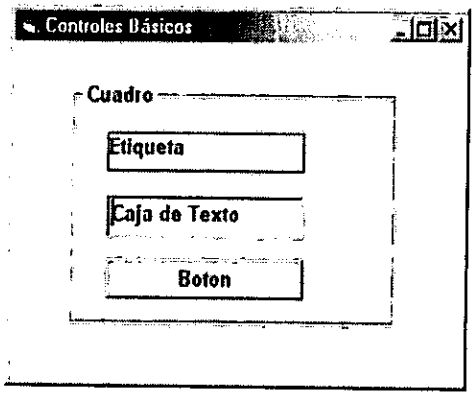
Ariel Guzmán 2001

21

Otros Controles.

Cuadros, (Frames).

Microsoft Visual Basic Ver. 6.0



Los cuadros permiten agrupar controles para que ésta se presente con claridad para el usuario final. Para colocar un control dentro de un cuadro es por medio de un click en el control que se desea dibujar en el cuadro y después dibujar el control con el mouse dentro del cuadro, de esta forma el control dibujado dentro del cuadro se liga directamente con este, es decir, si se mueve el cuadro, también se movera el control que se dibujó dentro de él.

Ariel Guzmán 2001

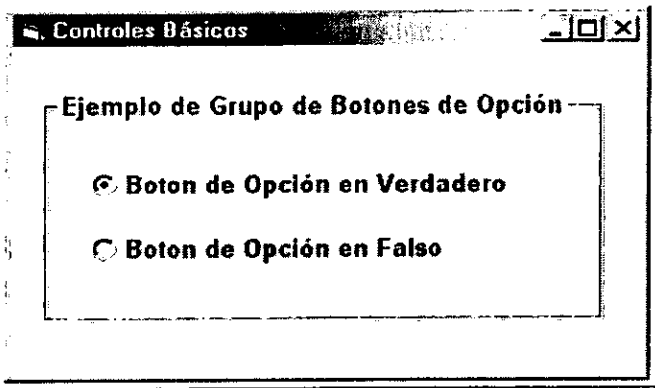
22

Otros Controles, (continuación).

Botones de Opción, (OptionButton).

Microsoft Visual Basic Ver. 6.0

Los botones de opción permiten elegir como su nombre lo indica, una opción de varias posibles. Es decir, estos botones son mutuamente excluyentes, de un grupo de botones, sólo se puede escoger uno y automaticamente los demás se quedan sin seleccionar.



Ariel Guzmán 2001

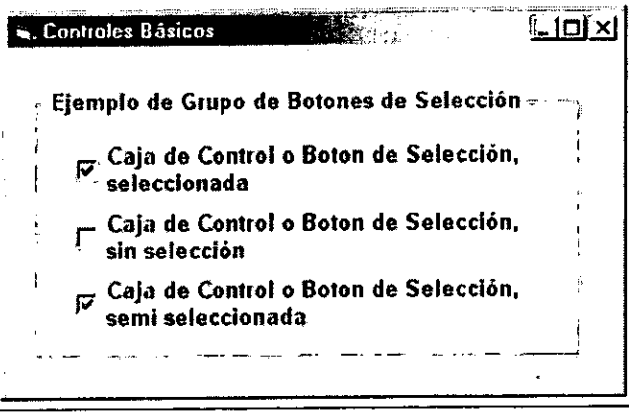
23

Otros Controles, (continuación).

Botones de Selección o Caja de Control, (CheckBox).

Microsoft Visual Basic Ver. 6.0

Los botones de selección permiten al usuario seleccionar de una serie de opciones, una o varias. Es decir, se pueden tener seleccionados más de uno y es lo que los diferencia de los botones de opción.



Ariel Guzmán 2001

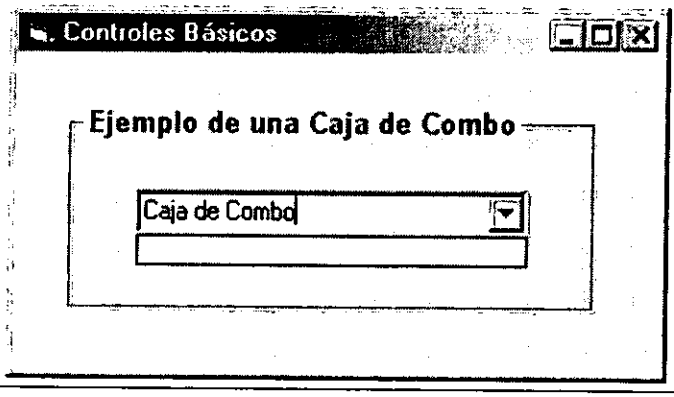
24

Otros Controles, (continuación).

Cajas de Combo, (ComboBox).

Microsoft Visual Basic Ver. 6.0

Estas cajas son muy habituales en el entorno de una aplicación y sirven para elegir un elemento que se presentra cuando se despliega la lista mediante la flecha que se encuentra a la derecha. Si hay muchos elementos puede tener una barra de desplazamiento vertical.



Ariel Guzmán 2001
25

Tipos de Datos en Visual Basic.

Variables y Constantes.

Microsoft Visual Basic Ver. 6.0

Variables: son valores que pueden cambiar durante la ejecución de la aplicación. Cuando se usa una variable en *Visual Basic*, se reserva un espacio en memoria para un valor que la aplicación requiere para funcionar o ser usadas en cálculos. Las variables pueden ser de diferentes tipos.

Constantes: son similares a las variables. Éstas se almacenan en memoria, pueden ser usadas en cálculos y pueden contener diferentes tipos de datos. Sin embargo, una constante no pueden ser cambiadas una vez definida.

Ariel Guzmán 2001
26

Tipos de Datos en Visual Basic, (continuación).

Tipos de Variables.

Microsoft Visual Basic Ver. 6.0

Tipo de Variable	Tamaño en Memoria	Carácter para declaración	Rango
Enteros (Integer)	2 bytes	%	- 32,768 a 32,767
Enteros Largos (Long Integer)	4 bytes	&	- 2,147,483,648 a 2,147,483,647
Precisión Simple	4 bytes	!	1.401298E-45 a 3.402823E38
Doble Precisión	8 bytes	#	4.94065645841247E-324 a 1.79769313486232E308
Cadena de caracteres (String)	10 bytes	$	0 a aprox. 2 millones (longitud variable)
Cadena de caracteres (String)	10 bytes	$	1 a aprox. 2 millones (longitud fija)
Variant (con números)	16 bytes	--	Cualquier valor numérico arriba de un Doble
Variant (con caracteres)	22 bytes	--	Mismo rango que para String (logitud variable)

Ariel Guzmán 2001

27

Tipos de Datos en Visual Basic, (continuación).

Declaración de Variables.

Microsoft Visual Basic Ver. 6.0

La manera en la cuál son declaradas las variables y donde ellas están declaradas determina como pueden ser usadas por la aplicación.

El comando "Option Explicit" exige la previa declaración de todas las variables para evitar un uso erróneo de las mismas.

Las variables pueden ser clasificadas por su alcance:

Locales, De módulo, Globales.

Ariel Guzmán 2001

28

Tipos de Datos en Visual Basic, (continuación).

Variables Locales.

Microsoft Visual Basic Ver. 6.0

Son las que solo tienen alcance dentro de un procedimiento (Sub, Función o Control).

Para declararlas se realiza utilizando el comando "DIM".

Dim x As Double, o bien, Dim x#

Ariel Guzmán 2001

29

Tipos de Datos en Visual Basic, (continuación).

Variables De Módulo.

Microsoft Visual Basic Ver. 6.0

Son las que tendrán alcance en todo un módulo, es decir, que valen lo mismo en todos los procedimientos de un módulo (Sub y Función).

Para declararlas se realiza utilizando el comando "DIM".

Dim x As Double, o bien, Dim x#

Ariel Guzmán 2001

30

Tipos de Datos en Visual Basic, (continuación).

Variables Globales.

Microsoft Visual Basic Ver. 6.0

Son aquellas accesibles en todo un proyecto, sin importar donde se encuentren.

Para declararlas se realiza utilizando el comando "Global".

Global x As Double

Ariel Guzmán
2001
31

Estructuras de Control.

Permiten a las aplicaciones escritas en *Visual Basic* a respondonder a diferentes situaciones dependiendo en los resultados de una ***condición***. La ***condición*** puede ser una comparación o cualquier expresión que evalua a un valor numérico.

Microsoft Visual Basic Ver. 6.0

Las estructuras de control disponibles son:

- Estructuras de Decisión
- Estructuras de Repetición o Bucles

Ariel Guzmán
2001
32

Estructuras de Decisión.

Los procedimientos de Visual Basic pueden probar condiciones y dependiendo de los resultados, realizar diferentes operaciones.

Microsoft Visual Basic Ver. 6.0

- Las estructuras de control disponibles son:

Sentencia If....Then
Sentencia If....Then....Else
Sentencia If....Then....ElseIf
Sentencia Select Case

•Operadores que pueden ser utilizados para evaluar una *condición*:

=	Igual
<>	Diferente
<	Menor que
>	Mayor que
<=	Menor o igual que
>=	Mayor o igual que

Ariel Guzmán 2001
33

Estructuras de Decisión, ***(continuación).***

Sentencia If....Then.

Microsoft Visual Basic Ver. 6.0

La sentencia ***If....Then*** evalúa si una ***condición*** es ***verdadera*** o ***falsa*** y por consiguiente sigue el flujo del programa. Se puede usar cualquiera de las dos sintaxis siguientes:

En una línea:

If condición ***Then*** *sentencia*

En varias líneas:

If condición ***Then***
sentencias
EndIf

Ariel Guzmán 2001
34

Estructuras de Decisión, (continuación).

Sentencia If....Then....Else

Microsoft Visual Basic Ver. 6.0

La sentencia ***If....Then....Else*** es una ampliación de la anterior. Permite seleccionar entre dos grupos de sentencias dependiendo de la ***condición***, y es muy útil para cuando es necesario evaluar más de una situación.

```
If condición Then
        sentencias
   Else
        sentencias
EndIf
```

Ariel Guzmán 2001

35

Estructuras de Decisión, (continuación).

Sentencia If....Then....ElseIf

Microsoft Visual Basic Ver. 6.0

La sentencia ***If....Then....ElseIf*** es como la anterior excepto que le permite al programa evaluar más de dos situaciones.

```
If condición A Then
        sentencias
   ElseIf condición B Then
        sentencias
   Else
        sentencias
EndIf
```

Ariel Guzmán 2001

36

Estructuras de Decisión, (continuación).

Sentencia Select Case.

Microsoft Visual Basic Ver. 6.0

La sentencia ***Select Case*** permite la selección de un conjunto de acciones a partir de una lista de diferentes opciones. Esta sentencia puede ser más eficiente porque evalúa la ***testexpresión*** sólo una vez. El resultado de la expresión es comparado contra los múltiples valores para determinar cual ***bloque de sentencias*** es llamado.

```
Select Case testexpresión
    Case expresión 1
        Bloque de sentencias 1
    Case expresión 2
        Bloque de sntencias 2
    Case Else
        Bloque de sentencias 'n'
End Select
```

Ariel Guzmán 2001

37

Estructuras de Repetición.

Las estructuras de repetición o bucle permiten ejecutar una o más líneas de código repetidamente.

Microsoft Visual Basic Ver. 6.0

Las estructuras de repetición disponibles son:

- Do....Loop
- For....Next

Ariel Guzmán 2001

38

Estructuras de Repetición, (continuación).

Sentencia Do...Loop.

Microsoft Visual Basic Ver. 6.0

Se utiliza para ejecutar un bloque de sentencias un número indefinido de veces. Hay algunas variantes en la sentencia ***Do...Loop***, pero cada una evalúa una condición numérica para determinar si continua la ejecución. Como ocurre con ***If...Then***, la ***condición*** debe ser un valor o una expresión que dé como resultado ***Falso*** (cero) o ***Verdadero*** (distinto de cero).

Ariel Guzmán
2001
39

Estructuras de Repetición, (continuación).

En el siguiente ejemplo de ***Do...Loop***, las ***sentencias*** se ejecutan siempre y cuando ***condición*** sea ***Verdadero***:

Microsoft Visual Basic Ver. 6.0

Do While *condición*
Sentencias
Loop

Cuando Visual Basic ejecuta este bucle ***Do***, primero evalúa ***condición***. Si ***condición*** es ***Falso*** (cero), se salta todas las sentencias. Si es ***verdadero*** (distinto de cero) Visual Basic ejecuta las ***sentencias***, vuelve a la instrucción ***Do While*** y prueba la condición de nuevo.

Ariel Guzmán
2001
40

Estructuras de Repetición, (continuación).

Variante de la instrucción ***Do...Loop***.

Microsoft Visual Basic Ver. 6.0

Ésta variante ejecuta las ***sentencias*** primero y prueba la ***condición*** después de cada ejecución. Ésta variación garantiza al menos una ejecución de sentencias:

Do
    Sentencias
Loop While *condición*

Ariel Guzmán 2001
41

Estructuras de Repetición, (continuación).

Otras variantes de la instrucción ***Do...Loop***.

Microsoft Visual Basic Ver. 6.0

Hay otras dos variantes análogas a las dos anteriores, excepto en que repiten el bucle siempre y cuando ***condición*** sea ***Falso*** en vez de ***Verdadero***.

Hace el Bucle cero o más veces.

Do Until *condición*
    Sentencias
Loop

Hace el Bucle al menos una vez.

Do
    Sentencias
Loop Until *condición*

Ariel Guzmán 2001
42

Estructuras de Repetición, (continuación).

Sentencia For...Next.

Microsoft Visual Basic Ver. 6.0

Los bucles ***Do*** funcionan bien cuando no se sabe cuántas veces se necesitará ejecutar las ***sentencias*** del bucle. Sin embargo, cuando se sabe el número determinado de veces que se va a ejecutar las ***sentencias***, es mejor elegir el bucle ***For...Next***. A diferencia del bucle ***Do***, el bucle ***For*** utiliza una variable llamada ***contador*** que incrementa o reduce su valor en cada repetición del bucle.

Ariel Guzmán
2001
43

Estructuras de Repetición, (continuación).

Sentencia For...Next.

Microsoft Visual Basic Ver. 6.0

For *contador = iniciar* ***To*** *finalizar [****Step*** *incremento]*
Sentencias
Next *contador*

Los argumentos ***contador***, ***iniciar***, ***finalizar*** e ***incremento*** son todos numéricos.

Nota: el argumento ***incremento*** puede ser positivo o negativo. Si ***incremento*** es positivo, ***iniciar*** debe ser menor o igual que ***finalizar*** o no se ejecutarán las sentencias del bucle. Si ***incremento*** es negativo, ***iniciar*** debe ser mayor o igual que ***finalizar*** para que se ejecute el cuerpo del bucle. Si no se utiliza ***Step***, el valor predeterminado de ***incremento*** es 1.

Ariel Guzmán
2001
44

Estructuras de Repetición, ***(continuación).***

Al ejecutar el bucle **For**, Visual Basic:

Microsoft Visual Basic Ver. 6.0

1. Establece ***contador*** al mismo valor que ***iniciar***.
2. Comprueba si ***contador*** es mayor que ***finalizar***. Si lo es, Visual Basic sale del bucle. (si ***incremento*** es negativo, Visual Basic comprueba si ***contador*** es menor que ***finalizar***.)
3. Ejecuta las ***sentencias***.
4. Incrementa ***contador*** en 1 o en ***incremento***, si se especificó.
5. Repite los pasos 2 a 4.

Ariel Guzmán
2001

45