UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

Facultad de Química

MICROCOMPUTADORAS Y METODOS NUMERICOS

T E S I S

Que Para Obtener el Título de

INGENIERO QUIMICO

P r e s e n t a :

EDUARDO VAZQUEZ ZAMORA

México, D. F., 1984

INDICE GENERAL

C A P I T U L O I

I.- INTRODUCCION

El objetivo de este trabajo es mostrar en una forma práctica y sintetizada los métodos numéricos que se involucran en la resolución de problemas de Ingeniería Química,así como también el uso de las microcomputadoras como herramientas de cálculo para atacar dichos problemas.

En este trabajo se plantean los principales métodos,sus características,en algunos casos sus ventajas y desventajas,su secuencia de cálculo,así como la cantidad de memoria utilizada para cada método empleando tres diferentes lenguajes de programación.

Al final de cada capítulo se plantean diferentes problemas representativos,analizando las alternativas que se tiene en cada método y el tiempo computacional.

C A P I T U L O I I

II.- GENERALIDADES

En general, los métodos y las técnicas de cálculo en ingeniería han cambiado considerablemente durante las dos últimas décadas con el advenimiento de la computadora digital, teniendo como resultado una marcada reducción en tiempo de cálculo y un incremento en la precisión computacional.

Consecuentemente, las computadoras no solo tienen las herramientas computacionales, sino que extienden el alcance de los cálculos en - problemas demasiado complejos, ésto tiene como resultado una revaluación en los métodos aplicables a determinados problemas.

Mientras estos incentivos son suficientes para justificar un amplio interés en las computadoras y su utilización en la solución - de problemas de Ingeniería Química; el Ingeniero Químico tiene ahora a su disposición elementos computacionales que le permiten manejar problemas de una complejidad tal que no podía ser manejados con las herramientas disponibles años atrás.

COMPARACION DE EQUIPOS DE CALCULO

A) Comparación de Computadoras.

Las computadoras modernas pueden ser clasificadas en digitales, analógicas e híbridas.

1.- Computadora digital.

La computadora digital tiene la característica de que las operaciones aritméticas tales como adición,multiplicación,división y-resta de valores discretos se lleva a cabo muy rápidamente,además-tiene gran capacidad de memoria lo que le permite el almacenamiento y la operación de un ilimitado número de valores discretos. Por lo tanto,las operaciones aritméticas pueden ser efectuadas con un alto grado de precisión,el cual es generalmente fijo,dependiendo de-la máquina que en particular se esté empleando.

Además de las operaciones artiméticas,la computadora digital - puede ser usada para tomar decisiones lógicas,por comparación entre sí,de valores discretos. Como resultado de estas características se tiene una herramienta computacional más versátil,la cuál cuando es unida con las técnicas numéricas como prueba y error,interpolación, etc. permite el manejo de una amplia variadad de problemas.

La programación de una computadora es una operación en la cuál se debe alimentar a la máquina una serie de instrucciones detalladas que ésta pueda entender y ejecutar,generalmente estas instrucciones se alimentan en un lenguaje altamente especializado.

Las computadoras digitales encuentran amplias aplicaciones en reducción de datos,estadística,solución de ecuaciones,etc.

Las técnicas numéricas y la versatilidad que acompaña a la computadora digital extienden su rango de aplicación.

2.- Computadora Analógica

En la computadora analógica,en contraste con la digital,los cálculos son efectuados en paralelo y contínuamente,ésto incluye a las operaciones aritméticas,así como también funciones contínuas como el caso de funciones trascendentales,etc.

La característica más importante de la computadora analógica es la capacidad de efectuar una integración en una base contínua. La solución de un determinado problema que presenta una computadora analógica está dada en forma contínua y es "análoga" a la solución exacta,no en términos de valores discretos los cuales pueden solamente aproximar la solución exacta. Otro punto es que los cálculos pueden ser llevados prácticamente a cualquier velocidad,ésto permite examinar rápidamente un amplio rango de parámetros en un espacio de tiempo que puede ser significativamente más corto que el tiempo fijo de cálculo en la digital.

La computadora analógica no posee la capacidad de almacenamiento que la computadora digital; además de que no tiene la capacidad de efectuar un gran número de operaciones sobre valores discretos.

Las computadoras analógicas encuentran su más grande aplicación en la solución de ecuaciones diferenciales,tal y como ocurre en - sistemas dinámicos en donde la cantidad de operaciones lógicas y- algebraicas requeridas es muy limitada.

La programación de una computadora analógica no requiere el conocimiento de un lenguaje altamente especializado como la digital. Los detalles de la programación son generalmente muy similares a - los métodos clásicos de solución de ecuaciones diferenciales.

3.- Computadora Híbrida.

Una computadora híbrida es la combinación de la computadora digital y de la computadora analógica en la cual,la solución de un - problema incorpora las ventajas de ambas. La porción de la computadora digital proporciona una gran capacidad de memoria y permite - operaciones lógicas y algebraicas,mientras que la parte de la analógica permite la integración contínua.

La transferencia de información de una porción a la otra y de - regreso,extiende la versatilidad de la híbrida. Las computadoras híbridas encuentran su aplicación en al solución de problemas complejos donde el tiempo requerido para una solución digital es excesivo y donde la analógica no posee la capacidad y la precisión requerida para una solución satisfactoria.

B) Microcomputadoras

Cuando las primeras computadoras fueron introducidas al mercado,el acceso a ellas lo tenían solamente las compañías que podían pagar los altos costos de renta y mantenimiento,pero con el avance de la electrónica y de la miniaturización,estos dispositivos se redujeron de tamaño,lo que permitió la introducción de las microcomputadoras hace pocos años,y consecuentemente,el acceso a ellas de parte de cualquier pequeña compañía o a tenerla incluso como computadora personal.

Una computadora puede ser dividida en cuatro elementos básicos como se muestra en la figura Nº 1.

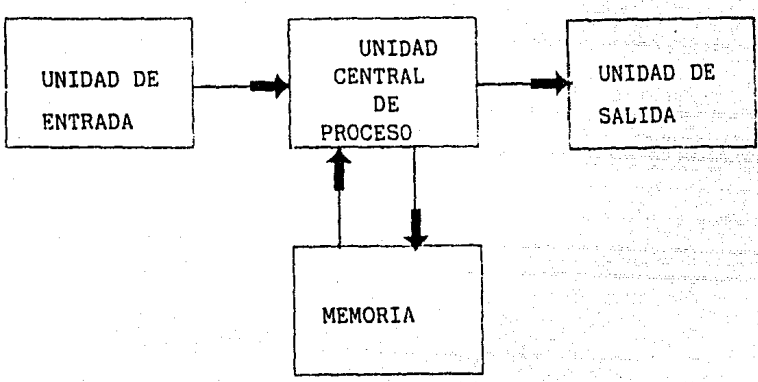
El primer elemento es la Unidad Central de Proceso comúnmente abreviado como CPU,la cual maneja todos los cálculos y controla los demás dispositivos conectados a ésta,ya que en realidad el CPU es el cerebro del sistema.

Para enviar información a la Unidad Central de Proceso es necesario conectar algún dispositivo de entrada al sistema; en las grandes computadoras uno de los dispositivos más comunes es la lectora de tarjetas,en las microcomputadoras,los teclados tipo máquina de escribir y las terminales CTR (Tubo de Rayos Catódicos) son los dispositivos de entrada más comunes.

La entrada de información al sistema es solamente la mitad de trabajo ya que también es necesario que la información ya procesada salga fuera de la máquina,esto es posible con dos dispositivos comunes que son,una impresora y una pantalla de video.

FIGURA Nº 1

ELEMENTOS BASICOS DE UNA COMPUTADORA DIGITAL



El último bloque en la figura Nº 1 es la memoria,que es un componente importante en cualquier sistema computacional,ésta permite a la computadora efectuar cálculos y almacenar resultados temporales para uso posterior.

La cantidad de memoria disponible tiene un gran efecto en la complejidad de los programas que pueden ser generados en la computadora,esto es,mientras mayor sea la cantidad de memoria,mayor complejidad se podrá tener en los programas que se manejen.

Todas las microcomputadoras disponibles hasta ahora,usan un circuito integrado llamado MICROPROCESADOR como la Unidad Central de Proceso (CPU),usualmente este cerebro contiene a la Unidad Lógica y Aritmética (ALU),en la cual se manejan operaciones lógicas y aritméticas.

El microprocesador es considerado el cerebro del sistema pero no le sería posible efectuar operaciones sin tener algo de memoria. Hay varios tipos de memoria usados en microcomputadoras,algunas de ellas contienen información e intrucciones para el microprocesador a este tipo de memoria se le conoce como Memoria de Lectura (ROM).

La principal ventaja de la memoria ROM es que nunca pierde la información almacenada aunque la potencia suministrada a la micro computadora se remueva.

Por esta razón,las compañías que construyen estos dispositivos computacionales proveen un conjunto de ROMS que contienen una serie de programas especiales,uno de ellos llamado MONITOR.

El monitor es un programa escrito en un tipo de lenguaje que solamente la máquina entiende; este programa es capaz de manejar algunas de las funciones elementales requeridas por el sistema,tales como el control de impresoras,memoria,operaciones de entrada y salida,etc.

No toda la memoria para una microcomputadora es del tipo ROM ya que en realidad mucha de la memoria debe tener la habilidad de ser fácilmente cambiada,este tipo de momoria es conocida como Memoria de Acceso Aleatorio (RAM),que puede ser cambiada y es normalmente usada como almacenamiento para programas,datos y almacenamiento temporal para variables usadas por el programa que esté en ejecución.

Los dos principales tipos de memoria RAM son: dinámicas y estáticas,las memorias dinámicas consumen menos potencia pero,su desventaja es que requieren de un ciclo de refrescado ó de otro modo pierden la información almacenada en éstas,en cambio las memorias estáticas no requieren ciclo de refrescado,pero su desventaja es que consumen mayor potencia.

La capacidad que tiene una microcomputadora de almacenar información,programas,etc.,en su memoria principal,generalmente se mide en los miles de caracteres ó bytes que ésta puede retener,- estas capacidades en microcomputadoras las encontramos en un amplio rango que van desde 2 KRAM hasta 128 KRAM; en realidad la - letra K no representa 1000,sino que representa en este caso 1024 caracteres y por lo tanto la capacidad real sería desde 2048 caracteres hasta 131072 en memoria RAM.

Por otro lado,no solamente en la memoria ROM se almacenan los programas monitores que permiten controlar equipo periférico conectado al CPU,sino que también se almacena en esta memoria el - lenguaje de alto nivel que facilita y acelera la preparación de - programas para la comunicación entre el usuario y la máquina. Generalmente esta capacidad de memoria ROM va de un rango de 8 KROM hasta 36 KROM,dependiendo del tipo de microcomputadora.

C) Lenguajes de Programación.

La programación de computadoras es una de las claves que forman el centro de esta nueva tecnología,tal y como ocurre en el campo de la computación en general,la tecnología de la programación ha experimentado muchos adelantos importantes,uno de ellos ha sido el desarrollo de lenguajes de programación orientados al usuario.

Tal y como dice el nombre,la programación,independientemente de la máquina,involucra el uso de lenguajes de computadora diseñados para reflejar la estructura propia del problema más que la estructura y la organización de un determinado computador.

Estos lenguajes se orientan a los problemas en el sentido de que ellos se asemejan muy de cerca al lenguaje y a las operaciones usadas para resolver los problemas de determinadas áreas.

Aunque los lenguajes de programación orientados a problemas están diseñados para ser independientes de la máquina; se requiere además,de un programa que traduzca estas instrucciones a un lenguaje que la máquina pueda entender.

Este tipo de programas se conocen con el nombre de compiladores e interpretadores. La diferencia principal que hay entre un compilador y un interpretador es que un compilador a partir de una serie de instrucciones escritas en un lenguaje de alto nivel las traduce a una serie de instrucciones que la máquina entiende,mientras que

el interpretador no traduce esa serie de instrucciones sino que efectúa una comparación de las instrucciones con las que tiene almacenadas en memoria ROM y sí es correcta la ejecuta,de otro modo,se detiene el proceso para corrección de la instrucción errónea.

En las microcomputadoras,la mayoria de ellas cuentan con interpretadores almacenados en memoria ROM. El lenguaje de alto - nivel almacenado es el BASIC (Codigo de Instrucción Simbólica para Principiantes).

El BASIC es el lenguaje de programación que recuerda las fórmulas elementales del algebra y debido a su simplicidad es ampliamente utilizado para resolver problemas científicos,de matemáticas e ingenieria.

Gracias a los avances tecnológicos es posible ahora disponer de compiladores en varios lenguajes de programación en microcomputadoras ya que solo antes estaban disponibles en sistemas de cómputo grandes,estos lenguajes son; FORTRAN,PASCAL,LISP,LOGO,PL/I,etc.;su disponibilidad depende principalmente del tipo de microcomputadora y de su configuración,como se aprecia en las tablas Nº 1 y 2.

TABLA Nº 1

CONFIGURACION MINIMA DE ALGUNAS MICROCOMPUTADORAS

MICROCOMPUTADORA	LENGUAJE	MEMORIA K RAM
TRS-80 mod. I	BASIC	4 - 16
APPLE II plus	APPLESOFT	16 - 48
ATARI 800	BASIC	16 - 48

TABLA Nº 2

CONFIGURACION MINIMA PARA USO DE COMPILADORES

MICROCOMPUTADORA	SISTEMA DE DISCO	MEMORIA K RAM *
TRS-80 mod. I	1	32
APPLE II plus	1	64
ATARI 800	1	48

* Mínimo

La microcomputadora utilizada en este trabajo para la elaboración de las subrutinas que se presentan en cada capítulo y que además se enlistan en tres lenguajes de programación que son: FORTRAN PASCAL y BASIC,éste último como interpretador,fué una microcomputadora APPLE II plus con 64 K RAM de memoria principal,una unidad de disco flexible de 5¼ pulgadas,con capacidad de 140,000 caracteres y una impresora con velocidad de impresión de 80 caracteres por segundo.

C A P I T U L O I I I

III.- METODOS PARA RESOLVER ECUACIONES NO LINEALES

A) Métodos discretos finitos

El desarrollo del análisis matemático moderno ha producido una nueva clase de métodos y procedimientos que son de gran ayuda para el científico,el ingeniero y el estudiante involucrados en las diferentes disciplinas de la ingeniería y de otras ramas.

Conocidos por el nombre general de métodos discretos finitos,la mayoría de estos nuevos métodos son conocidos por nombres específicos tales como: programación lineal,investigación de operaciones,- etc.

Lo importante de estos métodos es que emplean un número finito de pasos y operan sobre datos representados en forma discreta. Por esta razón,han provocado un aumento de interés por su aplicación a problemas científicos y de ingeniería con la ayuda de las - computadoras y más recientemente de las microcomputadoras.

Se han estado resolviendo un mayor número de problemas y a medida que se desarrollen nuevas aplicaciones de estos métodos,aumentarán las alternativas para atacar y resolver un número aún mayor- de problemas.

Los métodos desarrollados en este capítulo se utilizan en la solución de ecuaciones no-lineales,estos métodos con frecuencia encierran un gran número de cálculos que consumen mucho tiempo cuando se efectúan paso a paso y en forma individual,más aún en algunas circunstancias el ingeniero,el científico ó el estudíante pueden desear conocer el efecto que tendrán ciertos cambios en los parámetros del modelo sobre el comportamiento de un sistema,lo que involucra la solución de un mismo tipo de problema,muchas veces,con diferentes conjuntos de datos.

Estos métodos,también llamados de convergencia,son en realidad una solución por prueba y error de una ecuación implícita que involucra una unica variable;dicha solución consiste en suponer el valor que satisfaga la ecuación de la forma f(x)=0.

La precisión de estos métodos de convergencia solamente está limitada por el error de redondeo,el cual está determinado por un cierto número de lugares decimales,esto quiere decir que no es justificable llegar a una respuesta que tenga más decimales significativos que los datos de entrada.

Hay ecuaciones que se pueden resolver analíticamente en forma cerrada y que requieren gran cantidad de trabajo consecuentemente consumirán demasiado tiempo,este trabajo se puede eliminar programando estas ecuaciones en una computadora.

Hay otro tipo de ecuaciones que aunque se pueden resolver analíticamente se pueden obtener soluciones aproximadas empleando algún método de convergencia.

La primera fase en la solución de un problema por computadora,consiste en decidir sobre el método número rico que se va a utilizar para llegar a dicha solución.

Una vez seleccionado el método, el problema se debe programar en la forma de un proceso definido paso a paso,llamado algoritmo. Un algoritmo es una descripción precisa y completa de un proceso de - cálculo que garantiza la solución de un determinado problema,el cual puede incluir métodos números.

Además de la descripción paso a paso,generalmente se acostumbra definirlo en forma gráfica,mediante un diagrama de flujo,en el que los pasos se describen en forma de bloques,este diagrama nos muestra los pasos requeridos con gran detalle.

Los métodos de convergencia que se exponen en este capítulo y los demás métodos que aparecen en capítulos posteriores,se muestran en una forma muy práctica,que incluye además de sus características - su algoritmo,su diagrama de flujo y su enlistado de subrutinas elaboradas en tres lenguajes diferentes. Los métodos de convergencia que se exponen en este capítulo son:

1) Método de Bisección.
2) Método de Regula-Falsi.
3) Método de la Secante.
4) Método de Newton-Raphson..

1) Método de Bisección.

A este método también se le conoce como método del medio intervalo. La característica de este método es que a partir de un intervalo definido [X1,X2] ,éste se divide en dos y determina en que mitad se encuentra la raíz; este nuevo intervalo reducido,al que pertenece la raíz,vuelve a dividirse en dos y así sucesivamente hasta que el ancho del intervalo se haga menor que el error absoluto tolerado. Su representación gráfica se muestra en las figuras Nº 2 y 3.

Algoritmo

Característica de la función: f(X) = 0

1.- Acotar la función en un intervalo [X1,X2] tal que en él se halle una raíz o que al evaluar la función en dichos puntos sean de signo opuesto.

2.- Calcular el punto medio m como:

$$m = (X1 + X2)/ 2$$

3.- Aplicar el criterio de convergencia,evaluando la función en el punto medio o entre dos iteraciones sucesivas como:

$$| f(m) | \leq \text{tolerancia}$$

ò

$$| m_n - m_{n-1} | \leq \text{tolerancia}$$

si este criterio se cumple m es la aproximación a la raíz y el proceso se detiene.

4.- Evaluar la función en X1 y m para determinar en que mitad del intervalo se encuentra la raíz,aplicando los siguientes criterios:

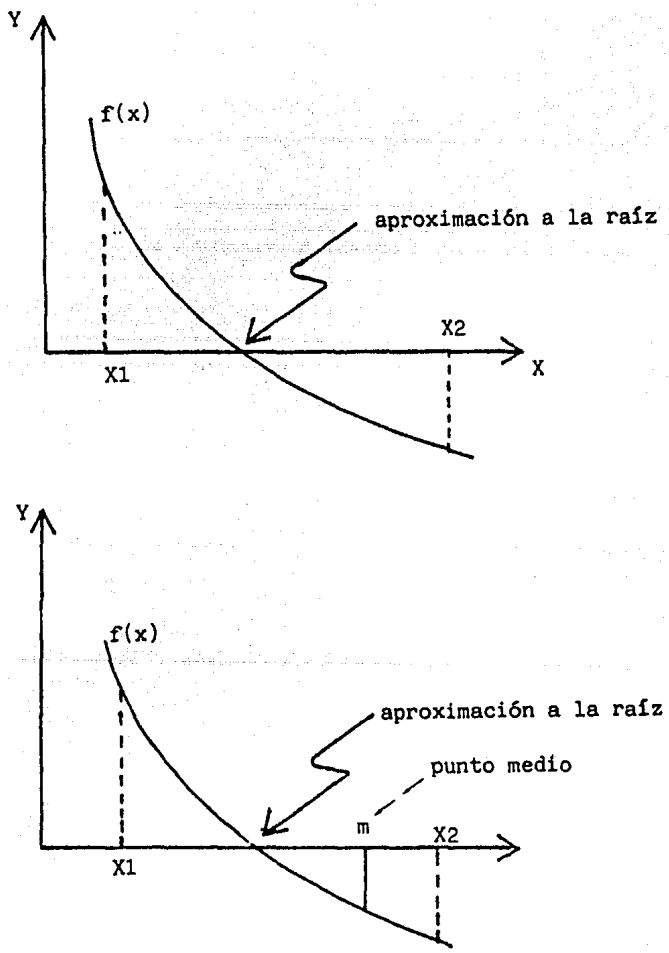
si $f(X1)*f(m) < 0$ asignamos $X1 = X1$ y $X2 = m$

si $f(X1)*f(m) > 0$ asignamos $X1 = m$ y $X2 = X2$

y regresamos al inciso 2 para continuar el proceso.

FIGURAS Nº 2 y 3

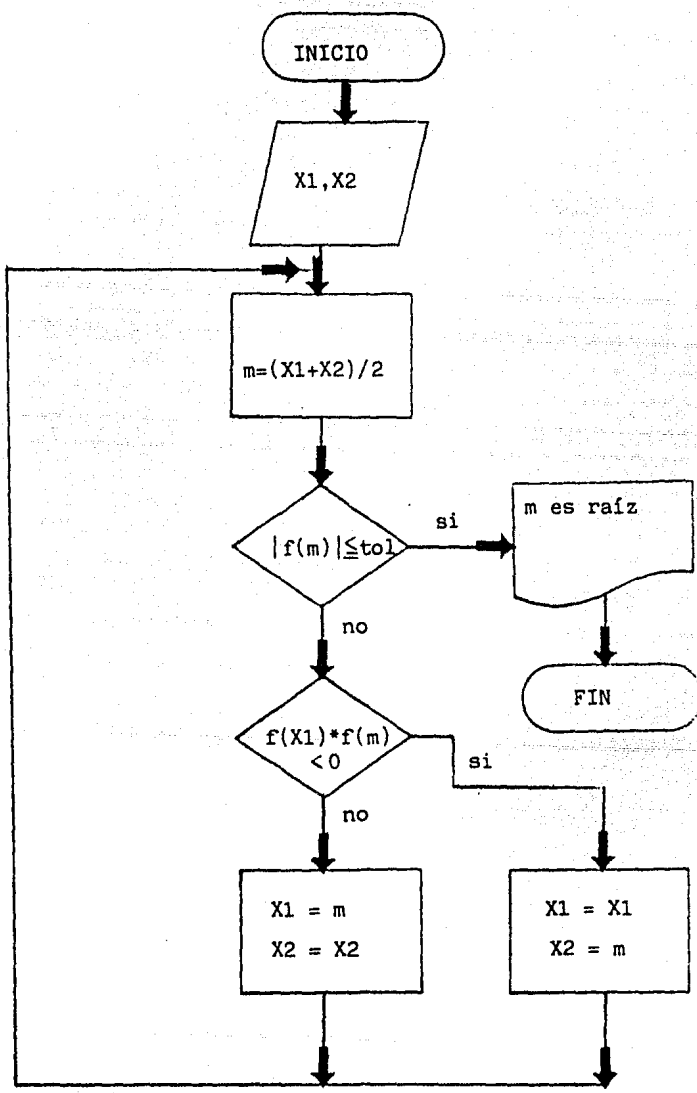
REPRESENTACION GRAFICA DEL METODO DE BISECCION



El diagrama de flujo se muestra en la figura Nº 4 para el método de bisección las subrutinas correspondientes se muestran en las siguientes hojas.

FIGURA Nº 4

DIAGRAMA DE FLUJO PARA EL METODO DE BISECCION



```
(*                                  *)
(* METODO DE MEDIO INTERVALO        *)
(* LENGUAJE:PASCAL                  *)
(*                                  *)
PROCEDURE MEDIO(VAR X1,X2,XN,EPS:REAL;VAR L:INTEGER);
 BEGIN
  IF FUNC(X1)*FUNC(X2) > 0.0THEN
    BEGIN
     L:=0;
     EXIT(MEDIO);
    END ELSE
      IF FUNC(X1)*FUNC(X2) = 0.0 THEN
       BEGIN
        L:=1;
        EXIT(MEDIO);
       END;
 FOR I:=1 TO 50 DO
   BEGIN
    XN:=(X1+X2)/2.0;
    IF ABS(FUNC(XN)) <= EPS THEN EXIT(MEDIO)
ELSE IF FUNC(X1)*FUNC(XN) < 0.0 THEN
    BEGIN
     X1:=X1;
     X2:=XN;
    END ELSE BEGIN
               X1:=XN;
               X2:=X2;
              END;
          END;
       L:=2;
   END;(*FIN *)
```

```
FORTRAN Compiler II.1  [1.1]
    0.      0 C
    1.      0 C *** METODO DE MEDIO INTERVALO ***
    2.      0 C
    3.      0 C ARGUMENTOS DE LA SUBRUTINA:
    4.      0 C 1) INTERVALO [X1,X2]
    5.      0 C 2) XN VALOR DE LA RAIZ
    6.      0 C 3) EPS TOLERANCIA
    7.      0 C 4) L=0 NO HAY RAIZ EN EL INTERVALO
    8.      0 C    L=1 ALGUNO DE LOS PUNTOS [X1,X2] ES RAIZ
    9.      0 C    L=2 NO SE ENCONTRO CONVERGENCIA
   10.      0 C        (50 ITERACIONES MAXIMO)
   11.      0 C
   12.      0       SUBROUTINE MEDIO(X1,X2,XN,EPS,L)
   13.      0       IF(FUNC(X1)*FUNC(X2))200,100,300
   14.     30   300 L=0
   15.     33       RETURN
   16.     35   100 L=1
   17.     38       RETURN
   18.     40   200 DO 150 N=1,50
   19.     48       XN=(X1+X2)/2.0
   20.     65       IF(ABS(FUNC(XN)).LE.EPS)RETURN
   21.     80       IF(FUNC(X1)*FUNC(XN).LT.0.0)GOTO 20
   22.    104       X1=XN
   23.    110       X2=X2
   24.    116       GOTO 150
   25.    118    20 X1=X1
   26.    124       X2=XN
   27.    130   150 CONTINUE
   28.    142       L=2
   29.    145       RETURN
   30.    147       END

ABS     INTRINSIC
EPS     REAL          2*
FUNC    REAL      FUNCTION      3,FWD
L       INTEGER       1*
MEDIO   SUBROUTINE              2
N       INTEGER       7
X1      REAL          5*
X2      REAL          4*
XN      REAL          3*
```

```
LIST

5000  REM
5010  REM  METODO DE MEDIO INTERVALO
5020  REM  LENGUAJE:BASIC
5030  IF  FN A(X1) *  FN A(X2) > 0 THEN L = 0: RETURN
5040  IF  FN A(X1) *  FN A(X2) = 0 THEN L = 1: RETURN
5050  FOR I = 1 TO 50
5060 X3 = (X1 + X2) / 2
5070  PRINT I;"    ";X3
5080  IF  ABS ( FN A(X3)) <  = 0.00001 THEN  RETURN
5090  IF  FN A(X1) *  FN A(X3) < 0 THEN 5130
5100 X1 = X3
5110 X2 = X2
5120  GOTO 5150
5130 X1 = X1
5140 X2 = X3
5150  NEXT I
5160 L = 2
5170  RETURN

]NEW

]
```

2) Método de Regula-Falsi.

Este método es más sofisticado que el anterior para determinar raíces,se basa en una interpolación lineal en cada iteración para determinar la aproximación a la raíz,además de escoger el intervalo donde se encuentra ésta.

Para aplicar este método la función debe estar en la forma f(x)=0 y acotarla en un intervalo $\lfloor XL,XR \rfloor$ tal que en éste,se encuentre una raíz. La figura Nº 5 muestra la representación gráfica de este método.

Algoritmo

Característica de la función: f(x)=0

1.- Acotar la función en un intervalo $\lfloor XL,XR \rfloor$ tal que,en este intervalo se halle una raíz ó bien que al efectuar f(XL)*f(XR) sean de signo opuesto.

2.- Calcular la aproximación a la raíz X_{n+1} con la siguiente expresión:

$$X_{n+1} = (XL*f(XR) - XR*f(XL))/(f(XR) - f(XL))$$

3.- Evaluar la función con X_{n+1} y aplicar el criterio de convergencia como sigue:

$$|f(X_{n+1})| \leq \text{tolerancia}$$

sí este criterio se cumple; X_{n+1} es la aproximación a la raíz y se detiene el proceso.

4.- Aplicar los criterios para determinar de que lado se encuentra la raíz:

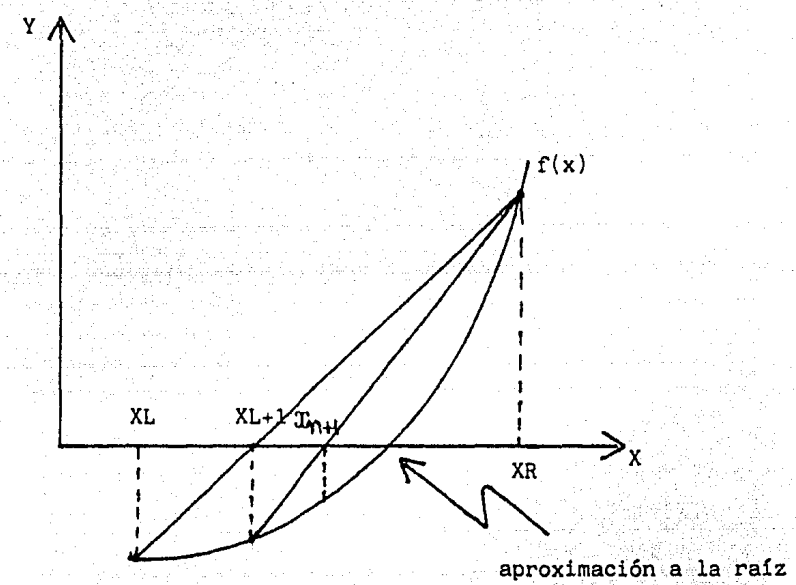
sí $f(X_{n+1})*f(XL) < 0$ asignar $XR = X_{n+1}$ y $XL = XL$

sí $f(X_{n+1})*f(XL) > 0$ asignar $XR = XR$ y $XL = X_{n+1}$

y regresamos al inciso 2 para continuar el proceso.

FIGURA Nº 5

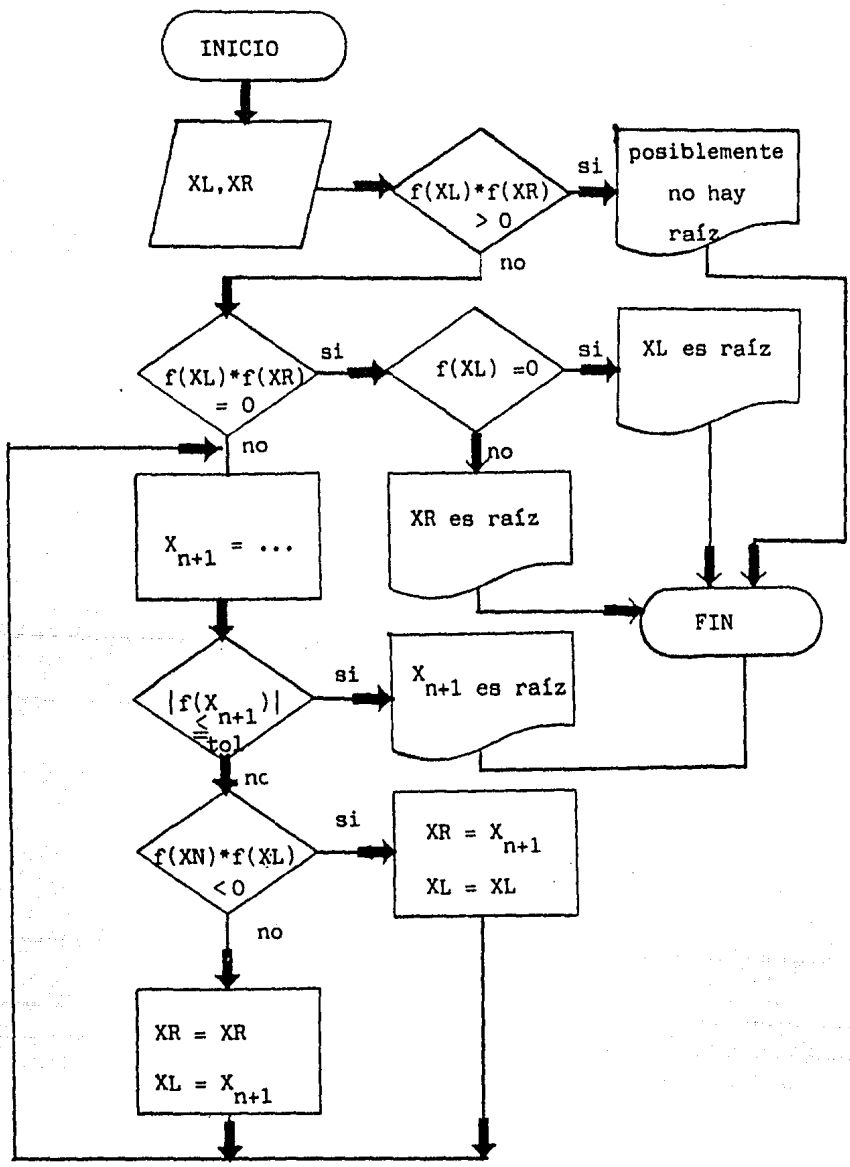
REPRESENTACION GRAFICA DEL METODO DE REGULA-FALSI



El diagrama de flujo para el método de Regula-Falsi se muestra en la figura Nº 6 y las subrutinas correspondientes en las siguientes hojas.

FIGURA Nº 6

DIAGRAMA DE FLUJO PARA EL METODO DE REGULA-FALSI



```
(*                                  *)
(* METODO DE REGULA FALSI           *)
(* LENGUAJE:PASCAL                  *)
(*                                  *)
PROCEDURE REGULA(VAR XL,XR,XN,EPS:REAL;VAR L:INTEGER);
 BEGIN
  IF FUNC(XL)*FUNC(XR) > 0.0 THEN
   BEGIN
    L:=0;
    EXIT(REGULA)
   END ELSE
  IF FUNC(XL)*FUNC(XR) = 0.0 THEN
     BEGIN
      L:=1;
      EXIT(REGULA);
     END;
FOR I:=1 TO 50 DO
 BEGIN
  XN:=(XL*FUNC(XR)-XR*FUNC(XL))/(FUNC(XR)-FUNC(XL));
  IF ABS(FUNC(XN)) <= EPS THEN EXIT(REGULA);
  IF FUNC(XN)*FUNC(XL) < 0.0 THEN
   BEGIN
      XL:=XL;
      XR:=XN;
   END ELSE BEGIN
            XL:=XN;
            XR:=XR;
             END;
          END;
     L:=2;
 END;(*FIN*)
```

```
FORTRAN Compiler II.1  [1.1]
    0.      0 C
    1.      0 C *** METODO DE REGULA-FALSI ***
    2.      0 C
    3.      0 C ARGUMENTOS DE LA SUBRUTINA:
    4.      0 C 1) INTERVALO [XL,XR]
    5.      0 C 2) XN VALOR DE LA RAIZ
    6.      0 C 3) EPS TOLERANCIA
    7.      0 C 4) L=0 NO HAY RAIZ EN EL INTERVALO
    8.      0 C    L=1 ALGUNO DE LOS PUNTOS [XL,XR] ES RAIZ
    9.      0 C    L=2 NO SE ENCONTRO CONVERGENCIA
   10.      0 C        (50 ITERACIONES MAXIMO)
   11.      0 C
   12.      0        SUBROUTINE REGULA(XL,XR,XN,EPS,L)
   13.      0        IF(FUNC(XL)*FUNC(XR))20,10,50
   14.     30     50 L=0
   15.     33        RETURN
   16.     35     10 L=1
   17.     38        RETURN
   18.     40     20 DO 95 N=1,50
   19.     48        XN=(XL*FUNC(XR)-XR*FUNC(XL))/(FUNC(XR)-FUNC(XL))
   20.     82        IF(ABS(FUNC(XN)).LE.EPS)RETURN
   21.     97        IF(FUNC(XN)*FUNC(XL).LT.0.0)GOTO 25
   22.    120        XL=XN
   23.    126        XR=XR
   24.    132        GOTO 95
   25.    134     25 XL=XL
   26.    140        XR=XN
   27.    146     95 CONTINUE
   28.    158        L=2
   29.    161        RETURN
   30.    163        END

ABS     INTRINSIC
EPS     REAL            2*
FUNC    REAL       FUNCTION      3,FWD
L       INTEGER         1*
N       INTEGER         7
REGULA  SUBROUTINE                2
XL      REAL            5*
XN      REAL            3*
XR      REAL            4*
```

```
LIST

5000  REM
5010  REM  METODO DE REGULA-FALSI
5020  REM  LENGUAJE:BASIC
5030  IF  FN A(X1) *  FN A(X2) > 0 THEN L = 0: RETURN
5040  IF  FN A(X1) *  FN A(X2) = 0 THEN L = 1: RETURN
5050  FOR I = 1 TO 50
5060 X3 = (X1 *  FN A(X2) - X2 *  FN A(X1)) / ( FN A(X2) -  FN A(X1))
5070  PRINT I;"    ";X3
5080  IF  ABS ( FN A(X3)) <  = 0.00001 THEN  RETURN
5090  IF  FN A(X1) *  FN A(X3) < 0 THEN 5130
5100 X1 = X3
5110 X2 = X2
5120  GOTO 5150
5130 X1 = X1
5140 X2 = X3
5150  NEXT I
5160 L = 2
5170  RETURN

]NEW

]
```

3) Método de la secante.

Tanto el método de Regula-Falsi,como el de Bisección requieren que se acote la función en dos puntos tales que al evaluar la función,estos valores sean de signo opuesto,para asegurar que hay -- raíz en ese intervalo.

El método de la secante es mucho más rápido que cualquiera de - los dos anteriores,utiliza la misma expresión que el de Regula-Falsi pero toma como valores las dos últimas estimaciones calculadas a la aproximación a la raíz. Sin embargo a cambio de su mayor rapidez,el método de la secante puede resultar inestable si las estimaciones - iniciales no están adecuadamente cercanas a la raíz buscada. Su representación gráfica se muestra en la figura Nº 7.

Algoritmo

Característica de la función: f(x)=0

1.- Acotar la función en un intervalo [XO,X1] tal que,en él se halle una raíz.

2.- Evaluar la aproximación a la raíz X_{n+1} con la siguiente expresión:

$$X_{n+1} = (X_{n-1} * f(X_n) - X_n * f(X_{n-1})) / (f(X_n) - f(X_{n-1}))$$

3.- Evaluar la función con la aproximación a la raíz X_{n+1} y aplicar el criterio de convergencia como:

$$|f(X_{n+1})| \leq \text{tolerancia}$$

sí se cumple,se detiene el proceso.

4.- Efectuar las siguientes asignaciones:

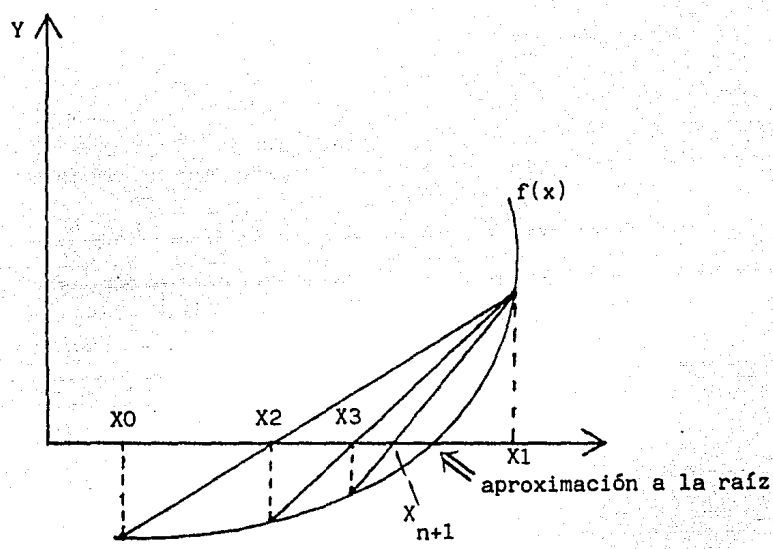
$$X_{n-1} = X_n$$

$$X_n = X_{n+1}$$

regresamos al inciso 2 para continuar el proceso.

FIGURA Nº 7

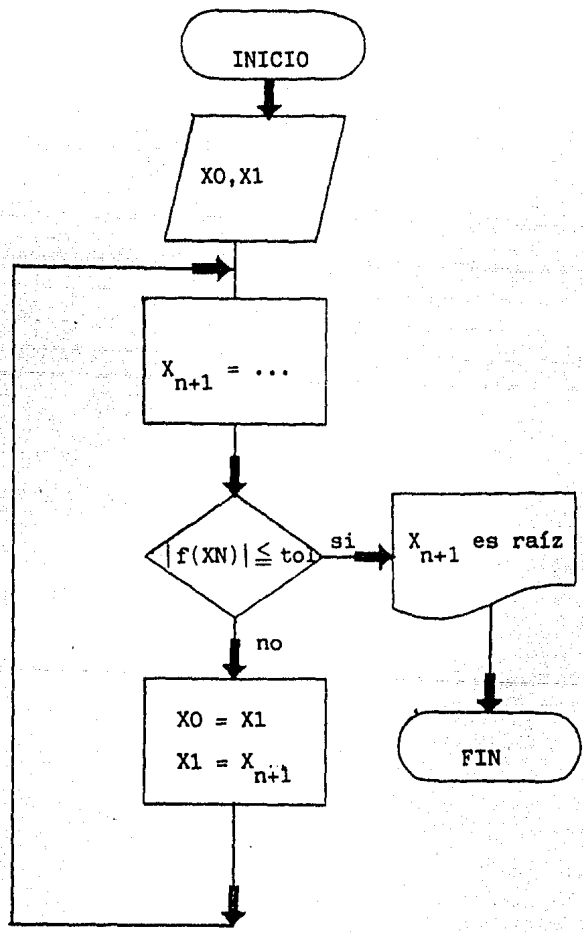
REPRESENTACION GRAFICA DEL METODO DE LA SECANTE



El diagrama de flujo del mètodo y las respectivas subrutinas se muestran a continuación.

FIGURA Nº 8

DIAGRAMA DE FLUJO DEL METODO DE LA SECANTE



```
FORTRAN Compiler II.1  [1.1]
    0.      0 C
    1.      0 C ###  METODO DE LA SECANTE ###
    2.      0 C
    3.      0 C ARGUMENTOS DE LA SUBFUTINA:
    4.      0 C 1) INTERVALO [X0,X1]
    5.      0 C 2) XN VALOR DE LA RAIZ
    6.      0 C 3) EPS TOLERANCIA
    7.      0 C 4) L=0 NO HAY RAIZ EN EL INTERVALO
    8.      0 C    L=1 ALGUNO DE LOS PUNTOS [X0,X1] ES RAIZ
    9.      0 C    L=2 NO SE ENCONTRO CONVERGENCIA
   10.      0 C        (50 ITERACIONES MAXIMO)
   11.      0 C
   12.      0 C LENGUAJE:FORTRAN
   13.      0 C
   14.      0       SUBROUTINE SECAN(X0,X1,XN,EPS,L)
   15.      0       IF(FUNC(X0)*FUNC(X1))90,45,105
   16.     30 105   L=0
   17.     33       RETURN
   18.     35 45    L=1
   19.     38       RETURN
   20.     40   90  DO 125 I=1,50
   21.     48       XN=(X0*FUNC(X1)-X1*FUNC(X0))/(FUNC(X1)-FUNC(X0))
   22.     82       IF(ABS(FUNC(XN)).LE.EPS)RETURN
   23.     97       X0=X1
   24.    103       X1=XN
   25.    109 125   CONTINUE
   26.    121       L=2
   27.    124       RETURN
   28.    126       END

ABS     INTRINSIC
EPS     REAL            2*
FUNC    REAL       FUNCTION      3,FWD
I       INTEGER         7
L       INTEGER         1*
SECAN   SUBROUTINE                2
X0      REAL            5*
X1      REAL            4*
XN      REAL            3*
```

```
(*                                    *)
(* METODO DE LA SECANTE               *)
(* LENGUAJE:PASCAL                    *)
(*                                    *)
PROCEDURE SECAN(VAR X0,X1,XN,EPS:REAL;VAR L:INTEGER);
 BEGIN
  IF FUNC(X0)*FUNC(X1) > 0.0 THEN
   BEGIN
    L:=0;
    EXIT(SECAN)
 END ELSE
     IF FUNC(X0)*FUNC(X1) = 0.0 THEN
       BEGIN
        L:=1;
        EXIT(SECAN)
       END;
FOR I:=1 TO 50 DO
 BEGIN
  XN:=(X0*FUNC(X1)-X1*FUNC(X0))/(FUNC(X1)-FUNC(X0));
  IF ABS(FUNC(XN)) <=  EPS THEN EXIT(SECAN);
     X0:=X1;
     X1:=XN
   END;
  L:=2;
END;(*FIN*)
```

```
LIST

5000  REM
5010  REM  METODO DE LA SECANTE
5020  REM  LENGUAJE:BASIC
5025  REM
5030  IF  FN A(X1) *  FN A(X2) > 0 THEN L = 0: RETURN
5040  IF  FN A(X1) *  FN A(X2) = 0 THEN L = 1: RETURN
5050  FOR I = 1 TO 50
5060 X3 = (X1 *  FN A(X2) - X2 *  FN A(X1)) / ( FN A(X2) -  FN A(X1))
5070  PRINT I;"    ";X3
5080  IF  ABS ( FN A(X3)) <  = 0.00001 THEN  RETURN
5090 X1 = X2
5100 X2 = X3
5110  NEXT I
5120 L = 2
5130  RETURN

]

]
```

4) Método de Newton-Raphson.

Este método es más rápido en la convergencia que los métodos anteriores, solamente que se necesita evaluar la derivada de la función y su desventaja es que ésta no debe tener máximos ni mínimos. Para aplicar el método se da un solo valor no importa que tan lejos esté de la aproximación de la raíz. Su representación gráfica y diagrama de flujo se muestran en las figuras Nº 9 y 10, las subrutinas correspondientes en las hojas siguientes.

Algoritmo

Característica de la función: f(x)=0 y f'(x)

1.- Dar un valor inicial XO.

2.- Evaluar la función y la derivada en XO.

3.- Calcular la nueva aproximación a la raíz con la siguiente expresión:

$$X_{n+1} = XO - f(XO)/f'(XO)$$

4.- Aplicar el criterio de convergencia como:

$$| X_{n+1} - XO | \leq \text{tolerancia}$$

sí se cumple lo anterior; X_{n+1} es la aproximación a la raíz y se detiene el proceso.

5.- Asignar:

$$XO = X_{n+1}$$

y regresar al inciso 2.

FIGURA Nº 9

REPRESENTACION GRAFICA DEL METODO DE NEWTON-RAPHSON

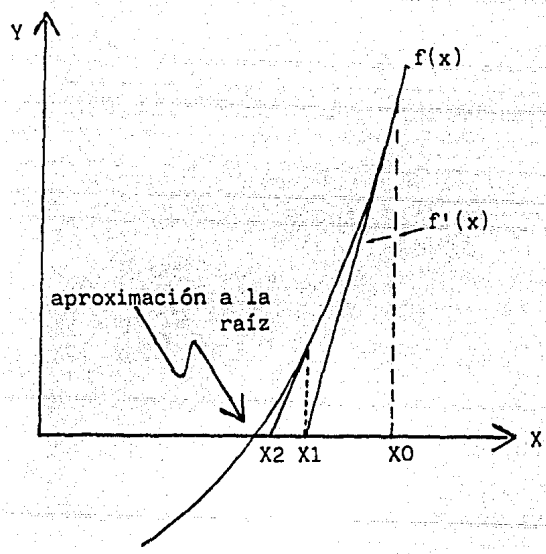
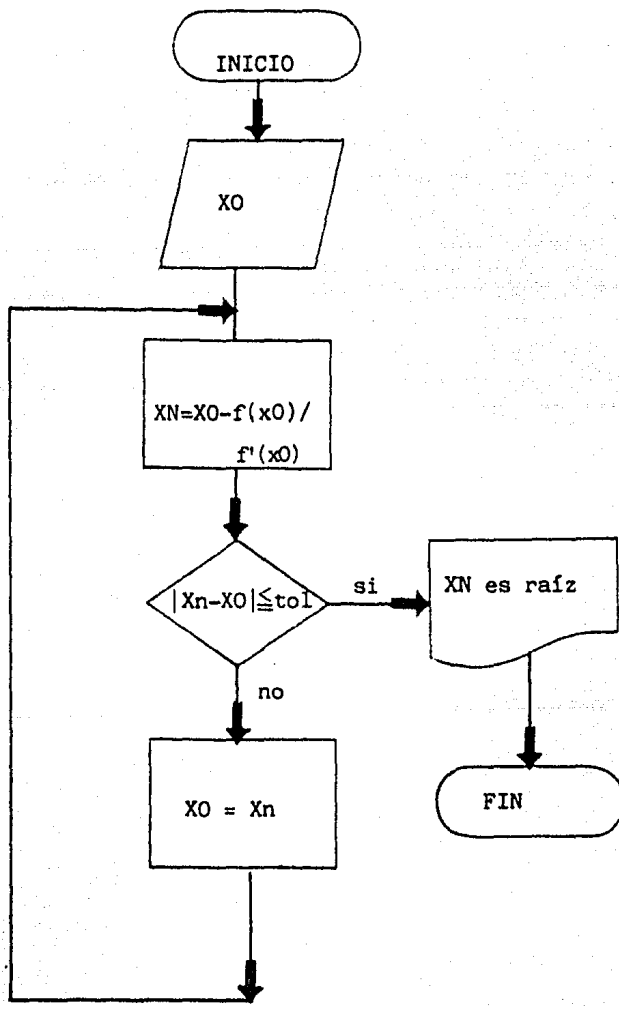


FIGURA Nº 10

DIAGRAMA DE FLUJO DEL METODO DE NEWTON-RAPHSON



```
FORTRAN Compiler II.1  [1.1]
    0.      0 C
    1.      0 C *** METODO DE NEWTON-RAPHSON ***
    2.      0 C
    3.      0 C ARGUMENTOS DE LA SUBRUTINA:
    4.      0 C 1) XO VALOR INICIAL
    5.      0 C 2) XN VALOR DE LA RAIZ
    6.      0 C 3) EPS TOLERANCIA
    7.      0 C 4) L=0 NO SE ENCONTRO CONVERGENCIA
    8.      0 C         (50 IERACIONES MAXIMO)
    9.      0 C
   10.      0       SUBROUTINE NEWTON(XO,XN,EPS,L)
   11.      0       L=1
   12.      3       DO 145 N=1,50
   13.     11       XN=XO-FUNC(XO)/DERIV(XO)
   14.     29       IF(ABS(XN-XO).LE.EPS)RETURN
   15.     46       XO=XN
   16.     52   145 CONTINUE
   17.     64       L=0
   18.     67       RETURN
   19.     69       END

ABS     INTRINSIC
DERIV   REAL       FUNCTION      4,FWD
EPS     REAL           2*
FUNC    REAL       FUNCTION      3,FWD
L       INTEGER        1*
N       INTEGER        6
NEWTON  SUBROUTINE                2
XO      REAL           4*
XN      REAL           3*
```

```
(*                                  *)
(* METODO DE NEWTON-RAPHSON         *)
(* LENGUAJE:PASCAL                  *)
(*                                  *)
PROCEDURE NEWTON(VAR TO,XN,EPS:REAL;VAR L:INTEGER);
 VAR
    XO:REAL;
 BEGIN
  L:=1;
  XO:=TO;
   FOR I:=1 TO 50 DO
      BEGIN
       XN:=XO-FUNC(XO)/DERIV(XO);
       WRITELN('APROXIMACION',' ':2,XN:10:3);
       IF ABS(XN-XO) <= EPS THEN EXIT(NEWTON)
       ELSE BEGIN
           XO:=XN;
         END;
      END;
   L:=0;
 END;(*FIN*)
```

```
 LIST

5000  REM
5010  REM  METODO DE NEWTON-RAPHSON
5020  REM  LENGUAJE:BASIC
5030  FOR I = 1 TO 50
5040 X3 = X1 -  FN A(X1) /  FN D(X1)
5050  PRINT I;"    ";X3
5060  IF  ABS (X3 - X1) <  = 0.00001 THEN  RETURN
5070 X1 = X3
5080  NEXT I
5090 L = 2
5100  RETURN

]NEW

]
```

B) Criterios de elección del método

Ningún método de los descritos anteriormente puede considerarse siempre superior a otro. Los criterios más comunes para comparar - algoritmos son: la seguridad en la convergencia,la rapidez de convergencia y la eficiencia computacional.

- El método de convergencia debe conducir a la raíz deseada de la ecuación.
- El método debe ser estable.
- El método debe conducir rápidamente a la solución deseada,demasiadas iteraciones ó muchos cálculos por iteración requieren más tiempo de computación.
- La tolerancia debe ser elegida de una manera tal que el valor de la aproximación a la raíz se encuentre dentro del grado de precisión deseado, pero no debe ser tan baja como para requerir un número de iteraciones innecesarias.
- La función debe ser reducida a cero y se puede escribir en - formas distintas,por medio de manipulaciones algebraicas.
- El rango de valores sobre el que puede variar la incógnita debe ser finito.
- La función f(x) no debe tener raíces imaginarias dentro del rango permisible de la incógnita.
- Los máximos y los mínimos y en menor escala los puntos de inflexión perjudican la convergencia.
- A medida que f(x) se hace aproximadamente lineal,la convergencia por cualquier método es más rápido.

C) PROBLEMAS DE APLICACION

Problema Nº 1

Las paredes de un horno están construidas por un tipo de ladrillo refractario de 15 cm. de espesor y conductividad térmica de 0.40 Kcal/m.hr.°C . Puesto en funcionamiento el horno,se observó que las pérdidas de calor al exterior eran muy grandes, y se procedió a aislarlo con una capa de 6 cm. de espesor de un material de conductividad térmica de 0.13 Kcal/m.hr.°C . Si en ambos casos la temperatura de la cara interna era de 1500 °C y la temperatura ambiente de 25°C y se desprecian las pérdidas de calor por radiación, calcular :

a) Temperatura de la cara externa no estando aislado.

b) Temperatura externa del aislante.

Experimentalmente se ha determinado que la temperatura de la cara externa no estando aislado viene dada por la siguiente expresión:

$$t_{ext} = t_i - E*A/Kc*(t_{ext} - t_{amb})^x$$

donde A y x son constantes propias de cada sistema en este caso:

DATOS:

A = 1.25 Kcal/hr

x = 1.25

t_i = 1500 °C

E = 0.15 m

Kc= 0.40 y 0.13 Kcal/m.hr. °C

t_{amb} = 25 °C

Para aplicar los métodos anteriores es necesario que la expresión tenga la forma f(x) = 0,en el caso de Newton-Raphson se requiere la derivada de nuestra función.

Función y derivada quedan de la siguiente manera:

$$f(x) : t_i - E*A/Kc*(t_{ext} - t_{amb})^{x} - t_{ext} = 0$$

$$f'(x) : -x*E*A/Kc*(t_{ext} - t_{amb})^{x-1} - 1$$

Los resultados de este problema se aprecian en las tablas Nº 3, 4 y 5, los cálculos fueron efectuados con una tolerancia de 0.00001.

TABLA Nº 3

LENGUAJE BASIC

METODO	ITERACION	TIEMPO*	INTERVALO	TEMPERATURA
Medio Intervalo	24	6.57	50,700	441.5289
Regula-Falsi	7	3.86	50,700	441.5289
Secante	5	2.44	50,700	441.5289
Newton-Raphson	4	1.67	700	441.5289
Newton-Raphson	5	1.91	50	441.5289

TABLA Nº 4

LENGUAJE PASCAL

METODO	ITERACION	TIEMPO*	INTERVALO	TEMPERATURA
Medio Intervalo	18	4.88	50,700	441.529
Regula-Falsi	5	3.09	50,700	441.529
Secante	4	2.12	50,700	441.529
Newton-Raphson	4	0.97	700	441.529
Newton-Raphson	4	1.12	50	441.529

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

TABLA Nº 5

LENGUAJE FORTRAN

METODO	ITERACION	TIEMPO*	INTERVALO	TEMPERATURA
Medio Intervalo	18	4.97	50,700.	441.529
Regula-Falsi	5	3.44	50,700	441.529
Secante	4	2.35	50,700	441.529
Newton-Raphson	4	1.10	700	441.529
Newton-Raphson	4	1.27	50	441.529

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

Problema Nº 2

Una bomba centrífuga es usada para transferir un líquido de un tanque a otro,como se muestra en la figura Nº 11 (ambos tanques están a un mismo nivel).

La bomba aumenta la presión del líquido de presión atmosférica P1 a una presión P2; este aumento de presión se pierde gradualmente debido a la fricción a lo largo del tubo, y a la salida se tiene - presión atmosférica P3.

El aumento de presión a traves de la bomba está dado por la siguiente relación empírica:

$$P2 - P1 = a - b*Q^{1.5}$$

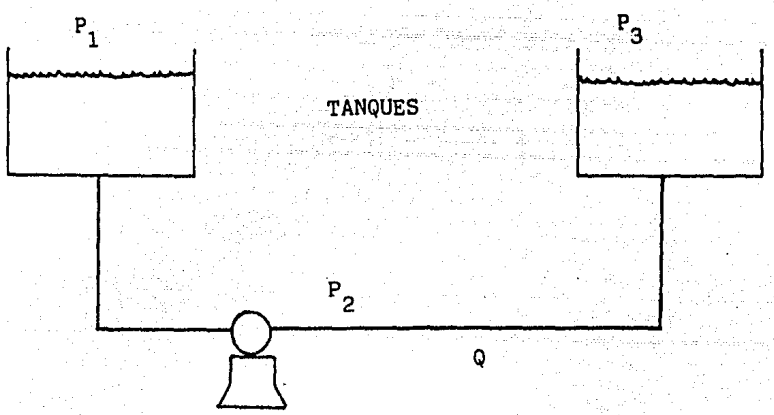
donde a y b son constantes que dependen de la bomba utilizada,Q es el flujo en gpm. La caida de presión en una tuberia horizontal de longitud L y diámetro interno d está dada por la siguiente expresión :

$$P2 - P3 = 2.16E-4*fm*ro*L*Q^{2}/D^{5}$$

donde ro es la densidad del fluido que está siendo bombeado. El factor de fricción fm se trata como constante,ya que realmente depende de la rugosidad de la tubería y del número de Reynolds.

El problema consiste en encontrar el flujo y la presión intermedia P2 y el flujo Q.

FIGURA Nº 11



DATOS:

a) Diámetro D = 2.469 in.
b) Longitud L = 210.6 ft.
c) Densidad ro = 51.4 lbm/ft^3
d) fm fm = 0.026 |a|
e) Cte. a a = 38.5 psi
f) Cte. b b = 0.0296 $psi/gpm^{1.5}$
g) Presión 1 P1 = 1.0 atm.
h) Presión 2 P2 = 1.0 atm.

Antes de aplicar los métodos anteriores es necesario efectuar algunas manipulaciones algebraicas para tener una expresión de la forma f(x) = 0,y para el método de Newton-Raphson se requiere la derivada f'(x).

Despejando de las expresiones anteriores las variables P1 y P3 tenemos lo siguiente:

$$(1)\ P2 = a - b*Q^{1.5} + P1$$

$$(2)\ P2 = 2.16E\text{-}4*fm*ro*L*Q^2/D^5 + P3$$

igualando las ecuaciones resultantes (1) y (2) y eliminando las presiones P1 y P3 debido a que ambas son iguales e igualando la ecuación a cero y derivando tenemos lo siguiente:

$$f(x) : 2.16E\text{-}4*fm*ro*L*Q^2/D^5 - a + b*Q^{1.5} = 0$$

$$f'(x) : 4.32E\text{-}4*fm*ro*L*Q/\ D^5 + 1.5*b*Q^{0.5}$$

Los resultados de este problema se observan en las tablas Nº 6,7 y 8.

TABLA Nº 6

LENGUAJE BASIC

METODO	ITERACION	TIEMPO*	INTERVALO	GASTO(gpm)
Medio Intervalo	20	12.78	11,160	103.898
Regula-Falsi	9	12.89	11,160	103.898
Secante	5	5.87	11,160	103.898
Newton-Raphson	5	2.06	160	103.898

TABLA Nº 7

LENGUAJE PASCAL

METODO	ITERACION	TIEMPO*	INTERVALO	GASTO(gpm)
Medio Intervalo	20	9.45	11,160	103.90
Regula-Falsi	9	10.05	11,160	103.90
Secante	5	4.27	11,160	103.90
Newton-Raphson	4	1.27	160	103.90

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

TABLA Nº 8

LENGUAJE FORTRAN

METODO	ITERACION	TIEMPO*	INTERVALO	GASTO(gpm)
Medio Intervalo	20	9.79	11,160	103.90
Regula-Falsi	9	10.23	11,160	103.90
Secante	5	4.71	11,160	103.90
Newton-Raphson	4	1.52	160	103.90

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

D) ANALISIS DE RESULTADOS

Como se observa en las tablas de resultados el método que podemos decir es más rápido en ejecución es el método de NEWTON-RAPHSON, el más lento es el método de BISECCION,ésto en los tres lenguajes BASIC,PASCAL y FORTRAN.

Otro punto importante que podemos observar es que el método de la SECANTE y de NEWTON-RAPHSON efectúan el mismo número de iteraciones pero en diferentes tiempos de ejecución.

Finalmente el tiempo de ejecución en FORTRAN es un poco mayor debido a que este lenguaje fué escrito en PASCAL.

C A P I T U L O I V

IV.- METODOS PARA RESOLVER SISTEMAS DE ECUACIONES LINEALES

Muchos problemas en ingeniería,y más precisamente en Ingeniería Química,involucran ecuaciones simultáneas lineales que describen - las relaciones entre dos o más variables.

En este capítulo se considera la solución de un conjunto de N - ecuaciones simultáneas lineales con N incógnitas. Estas ecuaciones son clasificadas como lineales puesto que cada término,en cada ecuación contiene solamente una incógnita,y cada incógnita aparece elevada a la primera potencia.

En general hay dos caminos para resolver un sistema de ecuaciones lineales y son: por técnicas de eliminación y por técnicas iterativas.

En principio,las técnicas de eliminación producen una solución - exacta en un número finito de operaciones aritméticas,mientras las- técnicas iterativas requieren un número infinito de operaciones aritméticas para producir una solución exacta.

Los métodos que se exponen en este capítulo son:

A) Método de eliminación de Gauss-Jordan.

B) Método iterativo de Gauss-Seidel.

A) Método de eliminación de Gauss-Jordan

La característica de este método consiste en combinar la matríz de coeficientes con el vector de términos independientes para formar la llamada "matríz aumentada", como ejemplo plantearemos un sistema de tres ecuaciones con tres incógnitas.

matríz de coeficientes

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

incógnitas

$$\begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix}$$

vector de términos independientes

$$\begin{Bmatrix} c_1 \\ c_2 \\ c_3 \end{Bmatrix}$$

matríz aumentada

$$\left|\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & c_1 \\ a_{21} & a_{22} & a_{23} & c_2 \\ a_{31} & a_{32} & a_{33} & c_3 \end{array}\right|$$

El método de eliminación de Gauss-Jordan consiste en reducir la matríz aumentada a la siguiente forma, donde la diagonal principal se encuentra formada por unos.

Matríz final despues de todas las eliminaciones :

$$\begin{vmatrix} 1 & 0 & 0 & c_1' \\ 0 & 1 & 0 & c_2' \\ 0 & 0 & 1 & c_3' \end{vmatrix}$$

la solución de nuestro sistema queda determinado a partir de las siguientes igualdades:

$$X_1 = c_1'$$
$$X_2 = c_2'$$
$$X_3 = c_3'$$

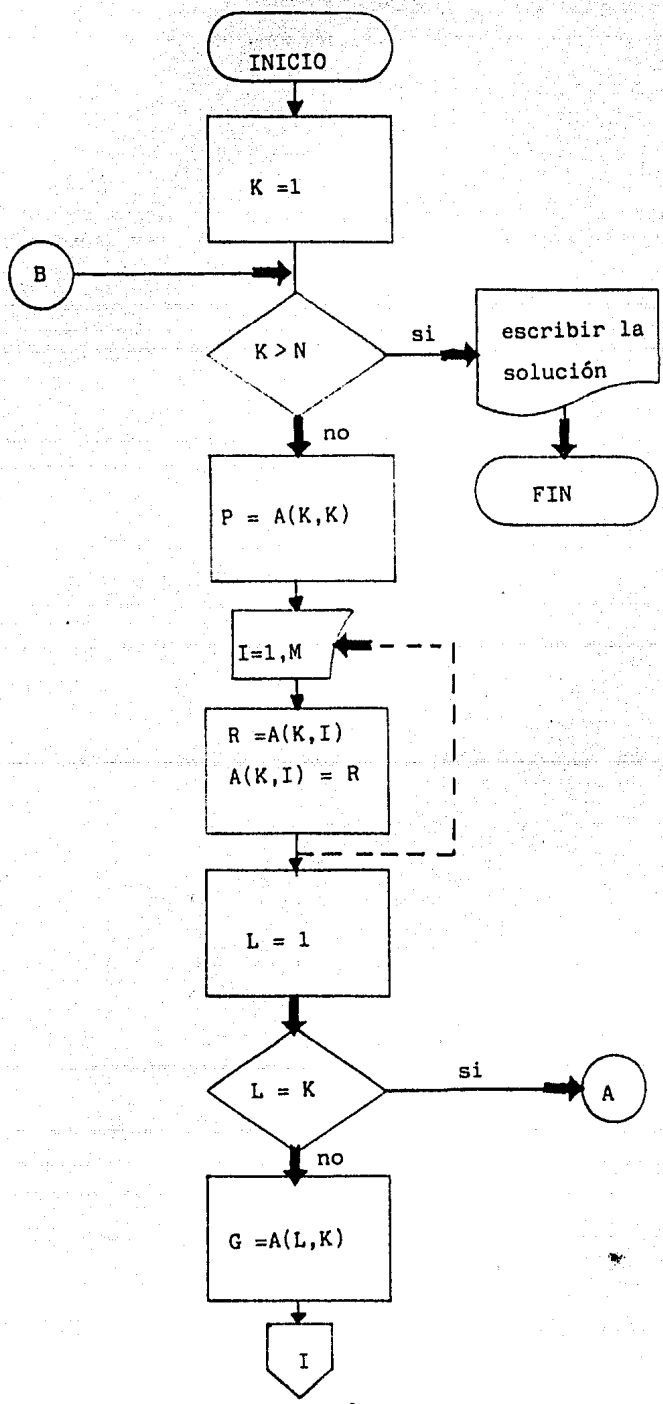
Algoritmo

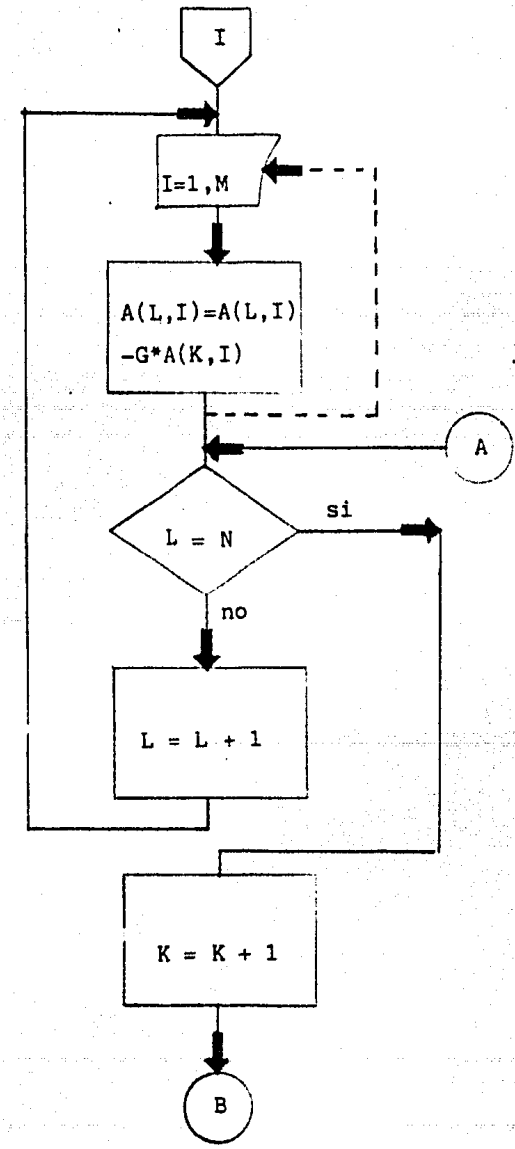
a) Intercambiar renglones de nuestra matríz original,si es necesario, para que la diagonal principal sea diferente de cero.
b) Tomar el elemento a_{ii} llamado pivote y dividirlo en todo el renglón (llamado renglón pivote) donde se halle éste.
c) Tomar el elemento que se desea eliminar a_{ij} y multiplicarlo por el renglón pivote.
d) Restar el renglón resultante del renglón donde se halla el elemento a_{ij} que se desea eliminar.
e) El proceso se continua desde el inciso (b) hasta llegar a la forma final de nuestra matríz.

En la figura Nº 12 se muestra el diagrama de flujo para este método,las subrutinas correspondientes en las hojas siguientes.

FIGURA Nº 12

DIAGRAMA DE FLUJO PARA EL METODO DE GAUSS-JORDAN




I
I=1,M
A(L,I)=A(L,I)
-G*A(K,I)
A
L = N
si
no
L = L + 1
K = K + 1
B

```
FORTRAN Compiler II.1  [1.1]
    0.      0 C
    1.      0 C *** METODO DE GAUSS-JORDAN ***
    2.      0 C
    3.      0 C 1) N=ORDEN DE LA MATRIZ
    4.      0 C 2) A=MATRIZ AUMENTADA A(N,N+1)
    5.      0 C 3) X=VECTOR DE RESULTADOS
    6.      0 C
    7.      0       SUBROUTINE GAUSS(N,A,X)
    8.      0       DIMENSION A(15,16),X(15)
    9.      0       M=N+1
   10.      6       DO 250 K=1,N
   11.     15       PIVOTE=A(K,K)
   12.     32       DO 770 I=1,M
   13.     40       REN=A(K,I)/PIVOTE
   14.     62       A(K,I)=REN
   15.     79   770 CONTINUE
   16.     91       L=1
   17.     94    99 IF(L.EQ.K)GOTO 350
   18.    101       G=A(L,K)
   19.    118       DO 23 I=1,M
   20.    126       A(L,I)=A(L,I)-G*A(K,I)
   21.    171    23 CONTINUE
   22.    183   350 IF(L.EQ.N)GOTO 250
   23.    191       L=L+1
   24.    196       GOTO 99
   25.    198   250 CONTINUE
   26.    210       DO 15 I=1,N
   27.    219    15 X(I)=A(I,M)
   28.    253       RETURN
   29.    255       END

A       REAL           2*
G       REAL          14
GAUSS   SUBROUTINE              2
I       INTEGER       10
K       INTEGER        6
L       INTEGER       13
M       INTEGER        4
N       INTEGER        3*
PIVOTE  REAL           7
REN     REAL          11
X       REAL           1*
```

```
(*                                *)
(* METODO DE GAUSS-JORDAN         *)
(* LENGUAJE:PASCAL                *)
(*                                *)
PROCEDURE JORDAN(VAR N:INTEGER;VAR A:MATRIZ;VAR X:VECTO);
 LABEL 1,2,3;
   VAR
     M,K,I,L:INTEGER;
     PIVOT,REN,G:REAL;
 BEGIN
  M:=N+1;
   FOR K:=1 TO N DO
    BEGIN
     PIVOT:=A[K,K];
     FOR I:=1 TO M DO
      BEGIN
       REN:=A[K,I]/PIVOT;
       A[K,I]:=REN;
      END;
    L:=1;
  3:IF L = K THEN GOTO 1;
 G:=A[L,K];
FOR I:=1 TO M DO
 A[L,I]:=A[L,I]-G*A[K,I];
  1:IF L = N THEN GOTO 2;
     L:=L+1;
      GOTO 3;
    2:END;
  FOR I:=1 TO N DO
   X[I]:=A[I,M];
END;(*FIN*)
```

```
LIST

5000  REM
5010  REM  METODO DE GAUSS-JORDAN
5020  REM  LENGUAJE:BASIC
5030  REM
5040 M = N + 1
5050  FOR K = 1 TO N
5060 P = A(K,K)
5070  FOR I = 1 TO M
5080 R = A(K,I) / P
5090 A(K,I) = R
5100  NEXT I
5110 L = 1
5120  IF L = K THEN 5170
5130 G = A(L,K)
5140  FOR I = 1 TO M
5150 A(L,I) = A(L,I) - G * A(K,I)
5160  NEXT I
5170  IF L = N THEN 5200
5180 L = L + 1
5190  GOTO 5120
5200  NEXT K
5210  FOR I = 1 TO N
5220 X(I) = A(I,M)
5230  NEXT I
5240  RETURN

]
```

B) Método iterativo de Gauss-Seidel

A diferencia del método de eliminación de Gauss-Jordan como método directo, el método de Gauss-Seidel es un procedimiento iterativo. Este procedimiento se basa en dar valores iniciales a las incógnitas X_i y calcular los nuevos valores por sustitución en nuestras ecuaciones; estos valores calculados son usados como nuevos estimados y se continúan las iteraciones hasta que algún criterio se cumpla para detener el proceso.

Consideremos el siguiente sistema de ecuaciones:

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \cdots\cdots\cdots + a_{1n}x_n &= c_1 \\
a_{21}x_1 + a_{22}x_2 + \cdots\cdots\cdots + a_{2n}x_n &= c_2 \\
a_{31}x_1 + a_{32}x_2 + \cdots\cdots\cdots + a_{3n}x_n &= c_3 \\
\vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad &\ \vdots \\
a_{n1}x_1 + a_{n2}x_2 + \cdots\cdots\cdots + a_{nn}x_n &= c_n
\end{aligned}
$$

Este sistema puede ser reescrito de tal manera que la i-ésima ecuación se resuelva para x_i.

Por lo tanto este sistema quedaría de la siguiente forma:

$$x_1 = -1/a_{11}(a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n) + c_1/a_{11}$$

$$x_2 = -1/a_{22}(a_{21}x_1 + a_{23}x_3 + \ldots + a_{2n}x_n) + c_2/a_{22}$$

$$\vdots$$

$$x_n = -1/a_{nn}(a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn-1}x_{n-1}) + c_n/a_{nn}$$

Antes de aplicar el algoritmo,se debe determinar si con nuestro sistema se tendrá convergencia,de la manera siguiente:

Si el valor absoluto del coeficiente dominante para una incógnita diferente en cada ecuación es mayor que la suma del valor absoluto de los coeficientes de esa ecuación la convergencia esta esegurada, esto es:

$$| a_{ii} | \geq \sum_{\substack{j \neq i \\ i = 1,2,3, \ldots N}} | a_{ij} |$$

Si lo anterior se cumple,debemos efectuar lo siguiente:

$$B_{ij} \begin{cases} -a_{ij}/a_{ii} & \text{para } i \neq j \\ 0 & \text{para } i = j \end{cases}$$

$$c_i = c_1/a_{ii}$$

Algoritmo

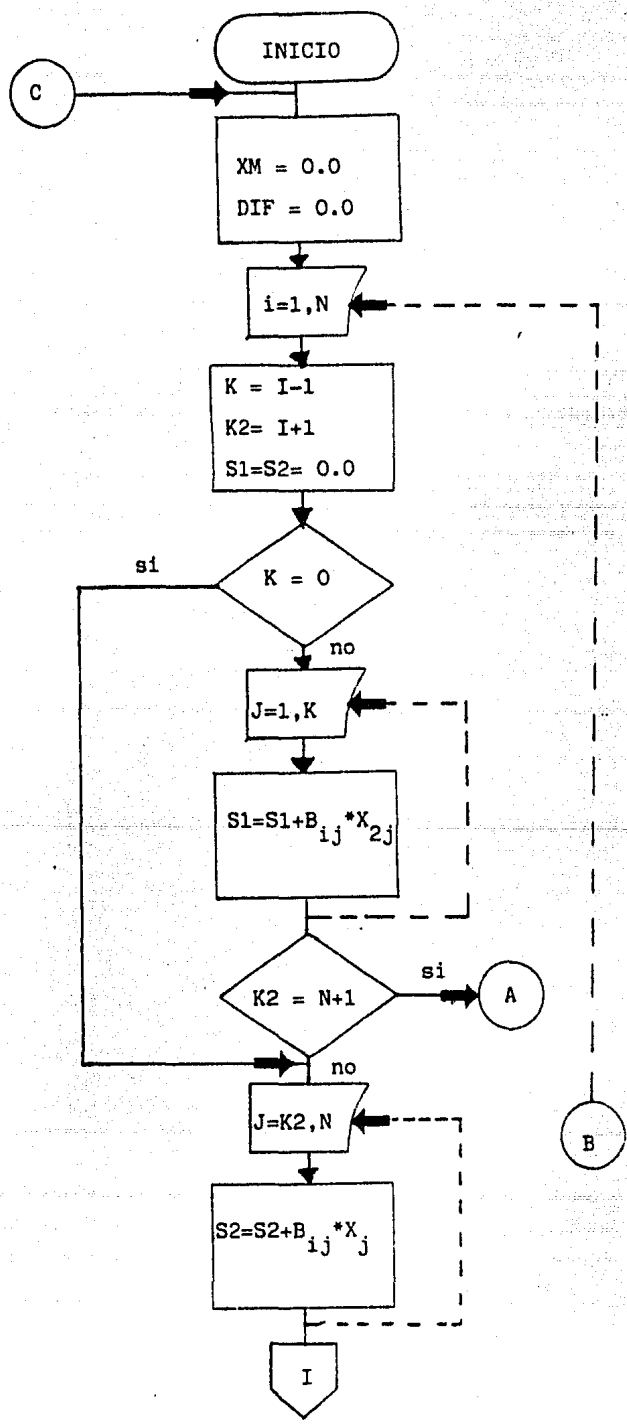
a) Escoger un valor inicial para las incógnitas,un buen valor de partida es dar un valor de cero a todas las incógnitas.
b) Esos valores se sustituyen en la ecuación i-ésima y se cálcula la nueva aproximación X_i.
c) Con esta aproximación X_i,se sustituye en la siguiente ecuación para calcular X_{i+1} y se continúa el proceso hasta que se tiene el nuevo vector de las incógnitas.
d) Este proceso se continúa desde el inciso (b) con los nuevos valores,para calcular los siguientes.
e) Se detiene el proceso aplicando el criterio de convergencia,que es entre dos aproximaciones sucesivas:

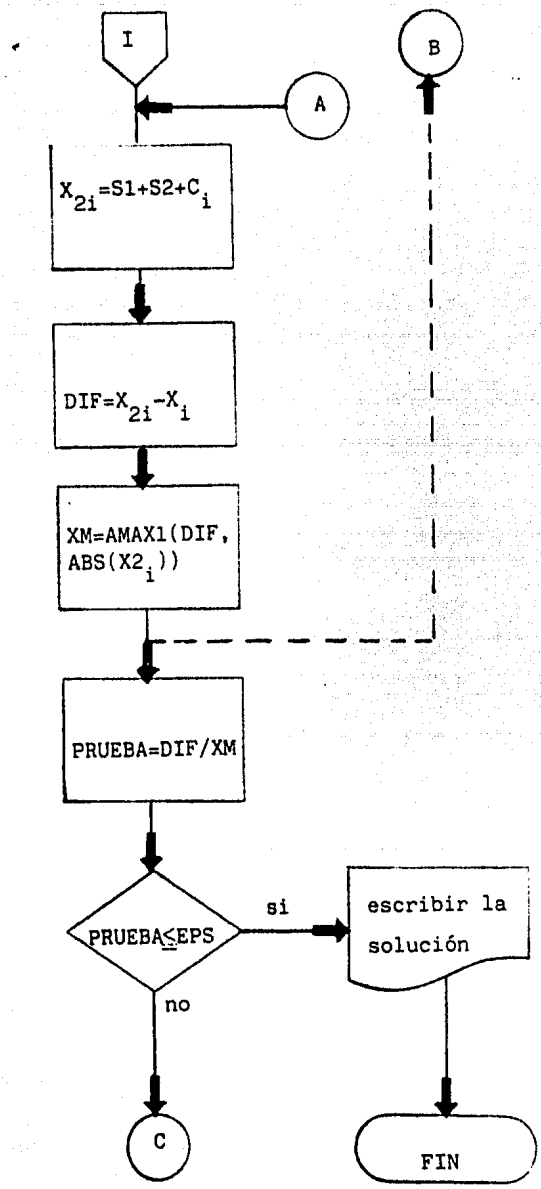
$$| (x_i^{k+1} - x_i^k) / x_i^k | \leq \text{precisión deseada}$$

El diagrama de flujo de este método se muestra en la figura Nº 13.

FIGURA Nº 13

DIAGRAMA DE FLUJO PARA EL METODO DE GAUSS-SEIDEL




I
A
B
X_{2i}=S1+S2+C_i
DIF=X_{2i}-X_i
XM=AMAX1(DIF, ABS($X2_i$))
PRUEBA=DIF/XM
PRUEBA≤EPS
si
escribir la solución
no
C
FIN

```
C
C ***  METODO DE GAUSS-SEIDEL ***
C
C ARGUMENTOS DE LA SUBRUTINA
C
C 1) N=ORDEN DE LA MATRIZ
C 2) A=MATRIZ AUMENTADA
C 3) EPS=TOLERANCIA
C 4) X=VECTOR DE SOLUCION Y VECTOR INICIAL
C
C LENGUAJE:FORTRAN
C
      SUBROUTINE SEIDEL(A,V,X,EPS,N)
      DIMENSION A(N,N),X(N),C(20),X1(20),B(20,20)
      DO 500 I=1,N
      DO 500 J=1,N
      IF(I.NE.J)GOTO 40
      B(I,J)=0.0
      GOTO 500
40    B(I,J)=-A(I,J)/A(I,I)
500   CONTINUE
      DO 14 I=1,N
14    C(I)=V(I)/A(I,I)
      M=0
600   M=M+1
      TOLE=0.0
      DO 60 I=1,N
      KX=I-1
      KXX=I+1
      SUMA1=0.0
      SUMA2=0.0
      IF(KX.EQ.0)GOTO 65
      DO 70 J=1,KX
70    SUMA1=SUMA1+B(I,J)*X1(J)
      IF(KXX.EQ.N+1)GOTO 5
65    DO 80 J=KXX,N
80    SUMA2=SUMA2+B(I,J)*X(J)
5     X1(I)=SUMA1+SUMA2+C(I)
      IF(ABS((X1(I)-X(I))/X1(I)).LE.TOLE)GOTO 34
      TOLE=ABS((X1(I)-X(I))/X1(I))
34    X(I)=X1(I)
60    CONTINUE
      IF(TOLE.LE.EPS)RETURN
14    IF(M.GE.60)GOTO 23
      GOTO 600
23    WRITE(6,12)
12    FORMAT(4X,'NO HAY CONVERGENCIA')
      RETURN
      END
```

```
(*                                *)
(* METODO DE GAUSS-SEIDEL         *)
(* LENGUAJE:PASCAL                *)
(*                                *)
PROCEDURE SEIDEL(VAR A:MAT;VAR V,X:VEC;EPS:REAL;VAR N:INTEGER);
 VAR
   M,I,J,KX,KXX:INTEGER;
   B:MAT;
   C,X1:VEC;
   SUMA1,SUMA2,TOLE:REAL;
    BEGIN
     FOR I:=1 TO N DO
     FOR J:=1 TO N DO
      BEGIN
       C[I]:=V[I]/A[I,I];
       IF I <> J THEN B[I,J]:=-A[I,J]/A[I,I]
       ELSE B[I,J]:=0.0
      END;
   M:=0;
   WHILE M < 60 DO
    BEGIN
     TOLE:=0.0;
      FOR I:=1 TO N DO
       BEGIN
        KX:=I-1;
        KXX:=I+1;
        SUMA1:=0.0;
        SUMA2:=0.0;
        IF KX = 0 THEN
          BEGIN
           FOR J:=KXX TO N DO
           SUMA2:=SUMA2+B[I,J]*X[J];
          END
   ELSE FOR J:=1 TO KX DO
           SUMA1:=SUMA1+B[I,J]*X1[J];
   IF KXX =N+1 THEN
      BEGIN
       X1[I]:=SUMA1+SUMA2+C[I];
       IF ABS((X1[I]-X[I])/X1[I]) <= TOLE THEN X[I]:=X1[I];
       TOLE:=ABS((X1[I]-X[I])/X1[I]);
      END;
     END;
  IF TOLE <= EPS THEN EXIT(SEIDEL);
     M:=M+1;
    END;
  WRITELN(' ':3,'NO HAY SOLUCION');
 END;(*FIN*)
```

```
LIST

5000  REM
5010  REM  METODO DE GAUSS-SEIDEL
5020  REM  LENGUAJE:BASIC
5030  REM
5040  FOR I = 1 TO N
5050  FOR J = 1 TO N
5060  IF I <  > J THEN 5090
5070 B(I,J) = 0.0
5080  GOTO 5100
5090 B(I,J) =  - A(I,J) / A(I,I)
5100  NEXT J,I
5110  FOR I = 1 TO N
5120 C(I) = V(I) / A(I,I)
5130  NEXT I
5140 M = 0
5150 M = M + 1
5160 T = 0.0
5170  FOR I = 1 TO N
5180 K1 = I - 1
5190 K2 = I + 1
5200 S1 = 0.0
5210 S2 = 0.0
5220  IF K1 = 0 THEN 5270
5230  FOR J = 1 TO K1
5240 S1 = S1 + B(I,J) * X1(J)
5250  NEXT J
5260  IF K2 = N + 1 THEN 5300
5270  FOR J = K2 TO N
5280 S2 = S2 + B(I,J) * X(J)
5290  NEXT J
5300 X1(I) = S1 + S2 + C(I)
5310  PRINT I; SPC( 5);X1(I)
5320  IF  ABS ((X1(I) - X(I)) / X1(I)) <  = T THEN 5340
5330 T =  ABS ((X1(I) - X(I)) / X1(I))
5340 X(I) = X1(I)
5350  NEXT I
5360  IF T <  = 0.001 THEN  RETURN
5370  IF M >  = 50 THEN  STOP
5380  GOTO 5150

]NEW

]
```

C) Criterios de elección del método

Es más eficiente manejar problemas en los cuales intervengan matrices densas (pocos elementos nulos) con métodcs directos como Gauss-Jordan, la desventaja de este método es que el número de eliminaciones necesarias para reducir una matríz es proporcional al cubo de su orden y los requerimientos de memoria para almacenar la matríz de coeficientes es proporcional al cuadrado de su orden.

El método de Gauss-Seidel se aplica a sistemas de ecuaciones grandes (generalmente de 20 o más elementos), con matrices de coeficientes esparcidas (varios elementos nulos), su desventaja es de que no siempre converge ó a veces converge muy lentamente.

D)PROBLEMAS DE APLICACION

Problema único

La alimentación a una unidad formada por dos columnas de destilación (Figura Nº 14) contiene 50% de un componente A, 30% de B, y 20% de C. La composición de las corrientes que salen de la unidad en una prueba de eficiencia fueron:

Componente	Destilado 1ª COL.	Destilado 2ª COL.	Residuo
A	4.5%	6.9%	95.5%
B	9.1%	90.1%	4.1%
C	86.4%	3.0%	0.4%

a) Se desea calcular para 1000 lb de alimentación la masa de cada uno de los destilados:

Solución

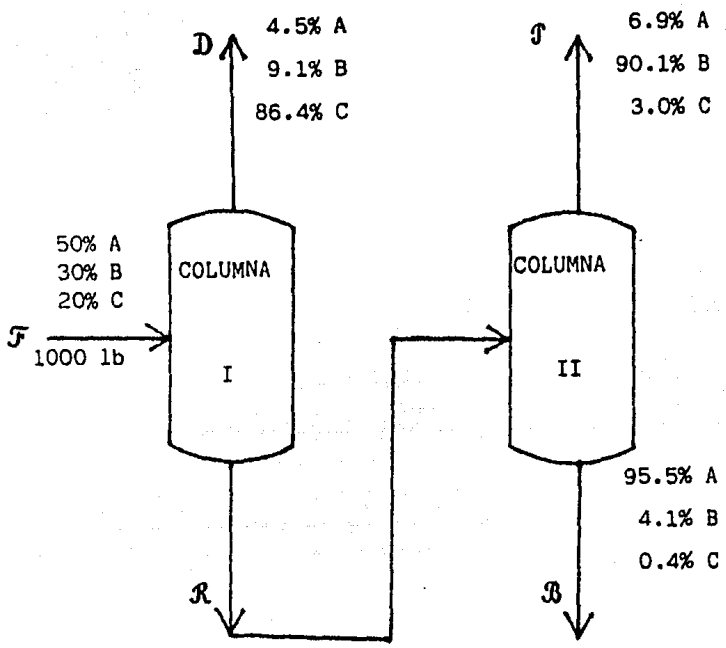
Antes de aplicar los métodos antes vistos,se efectua el balance de masa por componente:

A global :500 = 0.045D + 0.069P + 0.955B ...(1)
A torre I :500 = 0.045D : XaR ...(2)
B global :300 = 0.091D +0.901P + 0.041B ...(3)
B torre I :300 = 0.091D + XbR ...(4)

De las ecuaciones anteriores (1),(3) solamente se toman las que involucran D,P y B.

$$1000 = D + P + B \quad \text{(balance total)}$$
$$500 = 0.045D + 0.069P + 0.955B$$
$$300 = 0.091D + 0.901P + 0.041B$$

FIGURA Nº 14


D
4.5% A
9.1% B
86.4% C
T
6.9% A
90.1% B
3.0% C
50% A
30% B
20% C
F
1000 lb
COLUMNA
I
COLUMNA
II
95.5% A
4.1% B
0.4% C
R
B

Los resultados de este problema se aprecian en las tablas Nº 9,10 y 11.

TABLA Nº 9

LENGUAJE BASIC

METODO	ITERACIONES	TIEMPO*	RESULTADOS
GAUSS - JORDAN	-	1.13	D = 219.187323 P = 288.419342 B = 492.393336
GAUSS - SEIDEL	7	2.57	D = 219.187051 P = 288.419369 B = 492.393344

TABLA Nº 10

LENGUAJE PASCAL

METODO	ITERACIONES	TIEMPO*	RESULTADOS
GAUSS - JORDAN	-	0.92	D = 219.187 P = 288.420 B = 492.393
GAUSS - SEIDEL	7	1.09	D = 219.187 P = 288.420 B = 492.393

TABLA Nº 11

LENGUAJE FORTRAN

METODO	ITERACIONES	TIEMPO*	RESULTADOS
GAUSS - JORDAN	-	1.12	D = 219.187 P = 288.420 B = 492.393
GAUSS - SEIDEL	7	1.21	D = 219.187 P = 288.420 B = 492.393

* Tiempo medido en segundos.

* Promedio de 15 ejecuciones.

E) ANALISIS DE RESULTADOS

Podemos observar que en las tablas anteriores,el método de GAUSS-DEIDEL requiere de 7 iteraciones para llegar al resultado que se obtiene por el método de GAUSS-JORDAN.

En tiempos de ejecución el método que podemos decir que es "rápido" es el método de GAUSS-JORDAN,usando el lenguaje PASCAL,ya que entre los lenguajes BASIC y FORTRAN no hay diferencia.

Finalmente,en exactitud el método de GAUSS-JORDAN es el más exacto ya que el método de GAUSS-SEIDEL requiere de N iteraciones para llegar al resultado,además de que éste es una aproximación

C A P I T U L O V

V.- AJUSTE DE DATOS

En todos los campos de la ciencia,el trabajo experimental es la fuente de información más importante para el desarrollo de modelos y teorías que expliquen los fenómenos naturales;este trabajo experimental da como resultado un conjunto de valores de las propiedades del sistema que se esté estudiando.

La relación entre dos cantidades físicas llamadas "x" y "y",está descrita por una función $y = f(x)$,y es muy frecuente que la forma de esta función f(x) sea desconocida.

El estudio de la relación entre las dos cantidades "x" y "y" tiene dos aspectos fundamentales;el aspecto cuantitativo y el aspecto cualitativo.

Para el aspecto cuantitativo,nuestro problema consiste en calcular el valor de la función f(x) (cantidad física "y") con alguna -- "x" para la cual no se haya realizado el experimento.

El aspecto cualitativo consiste en lograr una descripción del comportamiento global del fenómeno, ésto no permite en un momento dado, deducir si por ejemplo la relación entre dos propiedades "x" y "y" - es lineal,exponencial o de cualquier otro tipo.

En muchos casos puede proponerse un modelo que trate de explicar el fenómeno. En este capítulo solamente se contempla el último aspecto que comúnmente es llamado Ajuste de Datos.

El algoritmo que se expone es usado para determinar:

a) La representación de un polinomio de grado M.

b) El grado del polinomio que adecuadamente representa a un conjunto de datos dado.

Algoritmo

El siguiente algoritmo estima los coeficientes del polinomio de grado M, representados de la siguiente forma:

$$Y_M = a_1 + a_2X + a_3X^2 + \ldots + a_{M+1}\,X^M$$

Esta técnica emplea algunos de los algoritmos expuestos en el capítulo anterior para la solución de sistemas de ecuaciones lineales, ya que para estimar los coeficientes $a_1, a_2, \ldots . a_{M+1}$ de un polinomio de grado M es necesario resolver un sistema formado de M+1 ecuaciones lineales con M+1 incógnitas, estas incógnitas son los coeficientes del polinomio de grado M.

a) Los pasos que se deben sequir cuando se tiene un polinomio de grado M fijo con N puntos $\{x_i\}$ y $\{y_i\}$ son los siguientes:

1.- Formar la matríz de coeficientes de nuestro sistema de ecuaciones llamadas también ecuaciones normales, de la siguiente manera:

$$\begin{aligned}
&a_1 N + a_2\textstyle\sum x + a_3\sum x^2 + \ldots + a_n\sum x^n = \sum y \\
&a_1\textstyle\sum x + a_2\sum x^2 + a_3\sum x^3 + \ldots + a_n\sum x^{n+1} = \sum xy \\
&\quad\vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \\
&a_1\textstyle\sum x^n + a_2\sum x^{n+1} + a_3\sum x^{n+2} + \ldots + a_n\sum x^{2n} = \sum x^n y
\end{aligned}$$

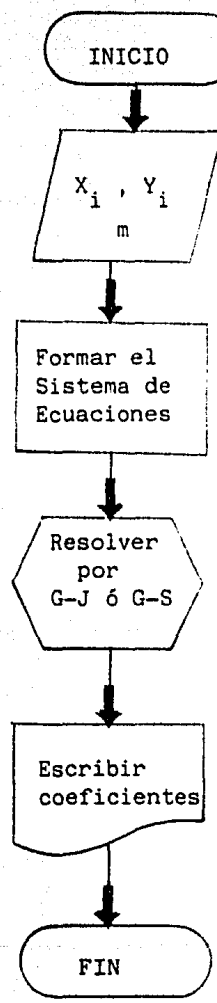
2.- Resolver el sistema de ecuaciones normales con alguno de los métodos del capítulo anterior para encontrar los coeficientes del polinomio.

3.- Formar el polinomio a partir de los coeficientes encontrados y detener el proceso.

El diagrama de flujo se muetra en la figura Nº 15 ,las subrutinas correspondientes en las siguientes hojas.

FIGURA Nº 15

DIAGRAMA DE FLUJO PARA DETERMINAR LOS COEFICIENTES DE UN POLINOMIO DE GRADO FIJO



b) Cuando se desea encontrar el polinomio de grado M que adecuadamente representa al conjunto de datos $\{X_i\}$ y $\{Y_i\}$, se debe hacer lo siguiente:

1.- Formar la matríz de coeficientes del sistema de ecuaciones normales para grado M = 1.

2.- Resolver el sistema de ecuaciones formado.

3.- Calcular el error total a partir del dato experimental y_i y el valor calculado a partir del polinomio encontrado, de la siguiente manera:

$$E_1 = \sum_{i=1}^{N} \left[y_i - (a_1 + a_2 x_i) \right]^2 / (N - M - 1)$$

4.- Formar la matríz de coeficientes del sistema de ecuaciones normales para grado M = 2.

5.- Resolver el sistema de ecuaciones formado.

6.- Calcular el error total como:

$$E_2 = \sum_{i=1}^{N} \left[y_i - (a_1 + a_2 x_i + a_3 x_i^2) \right]^2 / (N - M - 1)$$

7.- Determinar si $| E_1 - E_2 | \leq$ tolerancia; si ésto se cumple, se concluye que el mejor polinomio que representa al conjunto de datos es el polinomio de grado M = 1.

8.- Si lo anterior no se cumple incrementar el grado del polinomio, formar el sistema de ecuaciones normales, resolver y calcular el error total como:

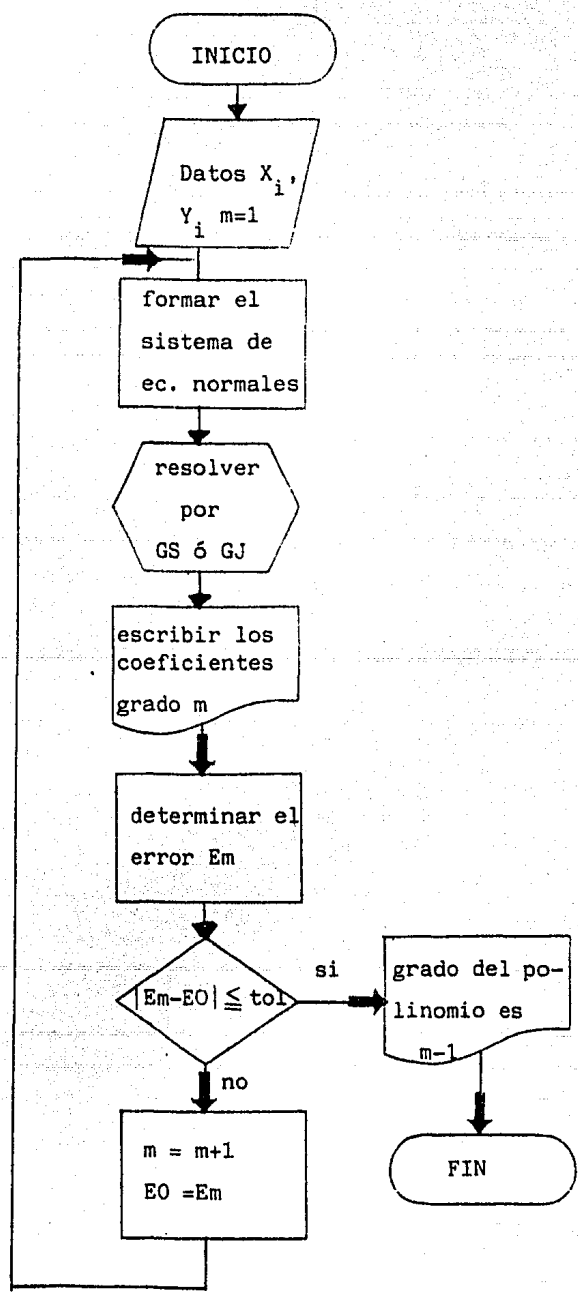
$$E_M = \sum_{i} \left[y_i - \left(\sum_{j} a_j x_i^{j-1} \right) \right]^2 / (N - M - 1)$$

9.- Determinar si $| E_M - E_{M-1} | \leq$ tolerancia; si ésto se cumple el polinomio que mejor representa a los datos, es el polinomio de grado M-1.

El diagrama de flujo se muestra en la figura Nº 16

FIGURA Nº 16

DIAGRAMA DE FLUJO PARA DETERMINAR LOS COEFICIENTES DE UN POLINOMIO DE GRADO VARIABLE



```
C
C ESTA SUBRUTINA PERMITE CONSTRUIR LA MATRIZ DE
C COEFICIENTES EN UN AJUSTE DE DATOS CUYA SOLUCION
C SON LAS CONSTANTES DE UN POLINOMIO DE GRADO 'N'
C
C PARAMETROS DE LA SUBRUTINA
C
C N=GRADO DEL POLINOMIO
C NDAT=NUMERO DE PAREJAS X,Y
C A=MATRIZ DE COEFICIENTES(MATRIZ AUMENTADA)
C
C LENGUAJE:FORTRAN
C
      SUBROUTINE MATRI(A,X,Y,N,NDAT)
      DIMENSION A(15,15),X(15),Y(15),S1(15),S2(15)
      N1=2*N
      N2=N+1
      S3=0.0
      S4=0.0
      DO 20 I=1,N
      N3=N+I
      S1(I)=0.0
      S1(N3)=0.0
      S2(I)=0.0
20    CONTINUE
      DO 35 I=1,NDAT
      S3=S3+Y(I)
      S4=S4+Y(I)**2
      D1=1.0
      DO 40 J=1,N
      D1=D1*X(I)
      S1(J)=S1(J)+D1
      S2(J)=S2(J)+Y(I)*D1
40    CONTINUE
      DO 45 J=N2,N1
      D1=D1*X(I)
      S1(J)=S1(J)+D1
45    CONTINUE
20    CONTINUE
      DO 99 I=1,N
      A(I,N2)=S2(I)-S3*S1(I)/FLOAT(NDAT)
      DO 110 I=1,N
      I1=I+J
      A(I,J)=S1(I1)-S1(I)*S1(J)/FLOAT(NDAT)
110   CONTINUE
99    CONTINUE
      RETURN
      END
```

```
(*                                                    *)
(* ESTA SUBRUTINA CONTRUYE LA MATRIZ DE               *)
(* COEFICIENTES DE LAS ECUACIONES NORMALES            *)
(* DE UN AJUSTE DE DATOS PARA LA DETERMINACION        *)
(* DE LAS CONSTANTES DE UN POLINOMIO DE GRADO         *)
(* N                                                  *)
(* NDAT=NUMERO DE PAREJAS X,Y                         *)
(* A=MATRIZ DE COEFICIENTES(MATRIZ AUMENTADA)         *)
(*                                                    *)
(* LENGUAJE:PASCAL                                    *)
(*                                                    *)
PROCEDURE MATRI(VAR A:MAT;VAR X,Y:VEC;VAR N,NDAT:INTEGER);
 VAR
   S1,S2:VEC;
   S3,S4,D1:REAL;
   N1,N2,N3,I,J,I1:INTEGER;
 BEGIN
  N1:=2*N;
  N2:=N+1;
  S3:=0.0;
  S4:=0.0;
   FOR I:=1 TO N DO
    BEGIN
     N3:=N+I;
     S1(I):=0.0;
     S2(I):=0.0;
     S1(N3):=0.0;
    END;
     FOR I:=1 TO NDAT DO
      BEGIN
       S3:=S3+Y[I];
       S4:=S4+Y[I]*Y[I];
       D1:=1.0;
        FOR J:=1 TO N DO
         BEGIN
          D1:=D1*X[I];
          S1[J]:=S1[J]+D1;
          S2[J]:=S2[J]+Y[I]*D1;
         END;
          FOR J:=N2 TO N1 DO
           BEGIN
         D1:=D1*X[I];
       S1[J]:=S1[J]+D1;
    END;
  END;
FOR I:=1 TO N DO
 BEGIN
 A[I,N2]:=S2[I]-S3*S1[I]/NDAT;
  FOR J:=1 TO N DO
   BEGIN
    I1:=I+J;
    A[I,J]:=S1[I1]-S1[I]*S1[J]/NDAT;
   END;
 END;
END;(*FIN*)
```

```
LIST

5000  REM
5010  REM  ESTA SUBRUTINA CONTRUYE LA MATRIZ DE
5020  REM  COEFICIENTES DE LAS ECUACIONES NORMALES
5030  REM  DE UN AJUSTE DE DATOS PARA LA DETERMINACION
5040  REM  DE LAS CONSTANTES DE UN POLINOMIO DE GRADO
5050  REM  N
5060  REM  Z=NUMERO DE PAREJAS X,Y
5070  REM  A=MATRIZ DE COEFICIENTES(MATRIZ AUMENTADA)
5080  REM
5090  REM  LENGUAJE:BASIC
5100  REM
5110 N1 = 2 * N
5120 N2 = N + 1
5130 S3 = 0.0
5135 S4 = 0.0
5140  FOR I = 1 TO N
5150 N3 = N + I
5160 S1(I) = 0.0
5162 S1(N3) = 0.0
5164 S2(I) = 0.0
5166  NEXT I
5170  FOR I = 1 TO Z
5180 S3 = S3 + Y1(I)
5190 S4 = S4 + Y1(I) ^ 2
5200 D1 = 1.0
5210  FOR J = 1 TO N
5220 D1 = D1 * X1(I)
5230 S1(J) = S1(J) + D1
5240 S2(J) = S2(J) + Y1(I) * D1
5250  NEXT J
5260  FOR J =`N2 TO N1
5270 D1 = D1 * X1(I)
5280 S1(J) = S1(J) + D1
5290  NEXT J,I
5300 F1 = Z
5310 C2 = S4 - S3 * S3 / F1
5320  FOR I = 1 TO N
5330 C1(I,1) = S2(I) - S3 * S1(I) / F1
5340 A(I,N2) = C1(I,1)
5350  FOR J = 1 TO N
5360 I1 = I + J
5370 A(I,J) = S1(I1) - S1(I) * S1(J) / F1
5380  NEXT J,I
5390  RETURN

]
```

A) PROBLEMAS DE APLICACION

Problema de aplicación único.

En esta unidad se determina el tiempo en resolver un polinomio de grado fijo y el tiempo en determinar el polinomio óptimo a partir de un conjunto de datos determinado.

Se desea determinar un polinomio que aproxime "Z" (factor de compresibilidad) como función de la Presión y Temperatura reducida a partir de los siguientes datos :

No.	Presión	Temperatura (1.20)
1	0.200	0.9590
2	0.600	0.8720
3	1.000	0.7800
4	1.400	0.6820
5	1.800	0.5920
6	2.200	0.5280
7	2.600	0.5200
8	3.000	0.5340
9	3.400	0.5590
10	3.800	0.5900
11	4.200	0.6240
12	4.600	0.6590
13	5.000	0.6940
14	5.400	0.7310
15	5.800	0.7690

Se desea determinar :

a) El polinomio de 4º grado.

b) Polinomio óptimo que mejor representa a los datos.

Los resultados de este problema se muestran en la tablas Nº 12 y 13.

TABLA Nº 12

POLINOMIO DE GRADO FIJO

LENGUAJE	TIEMPO*	CONSTANTES°	BASIC	PASCAL	FORTRAN
BASIC	10.69	Ao	1.0299	1.0299	1.0300
PASCAL	7.78	A1	-0.2861	-0.2861	-0.287
FORTRAN	8.35	A2	-2.5E-3	-2.5E-3	-2.5E-3
		A3	0.021	0.021	0.022
		A4	-2.3E-3	-2.3E-3	-2.3E-3

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

°Polinomio de 4° grado.

TABLA Nº 13

DETERMINACION DEL POLINOMIO DE GRADO OPTIMO

LENGUAJE	TIEMPO*	CONSTANTES°	BASIC	PASCAL	FORTRAN
BASIC	43.22	Ao	1.0059	1.0059	1.0059
PASCAL	34.63	A1	-0.2702	-0.2702	-0.2702
FORTRAN	38.87	A2	0.2175	0.2175	0.2175
		A3	-0.3180	-0.3180	-0.3180
		A4	0.1885	0.1885	0.1885
		A5	-0.0505	-0.0505	-0.0505
		A6	6.4E-3	6.4E-3	6.4E-3
		A7	-3.0E-4	-3.0E-4	-3.0E-4

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

°Polinomio óptimo de 7° grado.

B) ANALISIS DE RESULTADOS

Los tiempos que se observan en las tablas anteriores son los tiempos de ejecución,primero de la subrutina que nos forma el sistema de ecuaciones normales más el tiempo de ejecución de la subrutina que resuleve el sistema de ecuaciones.

Para determinar las constantes de un polinomio de 7º grado se requieren 34.63 segundos,mientras que para uno de 4º grado solamente se requiere de aproximadamente 8 segundos,lo que podemos concluir es de que a medida de que se incrementa el grado del polinomio,el tiempo de ejecución se verá incrementados notablemente.

En el lenguaje PASCAL este tiempo de ejecución es el que resulta más razonable.

C A P I T U L O V I

VI.- METODOS PARA RESOLVER ECUACIONES DIFERENCIALES DE PRIMER ORDEN

Muchos procesos físicos,particularmente sistemas dependientes del tiempo,pueden ser descritos por ecuaciones diferenciales ordinarias.

Estos métodos de solución para estas ecuaciones son de gran importancia para ingenieros y científicos.

Muchas ecuaciones diferenciales pueden ser resueltas por técnicas analíticas,pero un gran número de ecuaciones diferenciales no pueden ser resueltas; afortunadamente,la solución para estas ecuaciones puede ser generada numéricamente.

En este capítulo se exponen los métodos para la solución de ecuaciones diferenciales ordinarias de primer orden; llamadas así porque su más alta derivada está elevada a la primera potencia y ordinaria porque solamente las derivadas totales aparecen en la ecuación.

Esta solución,generada numéricamente,está dada en forma tabular para n+1 valores discretos de X,esta tabla de valores contiene una aproximación particular de la solución de la ecuación en un intervalo sobre el cual una solución es deseada,este intervalo [a,b] generalmente se divide en subintervalos ó pasos.

Los métodos de solución varían en complejidad,ya que en general, a mayor precisión de un método corresponde mayor complejidad,los métodos que se exponen en este capítulo en forma muy simple son:

A) Método de Eüler.

B) Método de Runge-Kutta.

En general los métodos que requieren únicamente del conocimiento de Yn para determinar Yn+1 son conocidos como métodos de arranque, de un escalon ó seccionalmente escalonados. Para aplicar los métodos anteriores es necesario conocer la condiciones iniciales para la ecuación diferencial (XO,YO).

A) Método de EUler

Este método es uno de los más simples para la aproximación de la solución a una ecuación diferencial con condiciones iniciales conocidas (XO,YO),esta solución es generada en pequeños incrementos en nuestra variable independiente X,este pequeño incremento se le conoce como tamaño de paso y se determina como h = (b - a)/n donde[a,b] es el intervalo en el cual se desea la solución y n es el número de subintervalos en el que se divide [a,b].

Este método se utiliza para problemas prácticos en donde no se requiere demasiada precisión.

Algoritmo

1.- Conocer las condiciones iniciales XO,YO.

2.- Fijar el tamaño de paso h y X_{final}

3.- Calcular el nuevo valor de Y_{i+1} a partir de la siguiente expresión:

$$Y_{i+1} = Y_i + h*f(X_i,Y_i)$$

4.- Incrementar la variable independiante X como:

$$X_{I+1} = X_i + h$$

5.- Detener el proceso si se cumple lo siguiente:

$$X_{i+1} \geqq X_{final}$$

6.- Asignar:

$$X_i = X_{i+1}$$

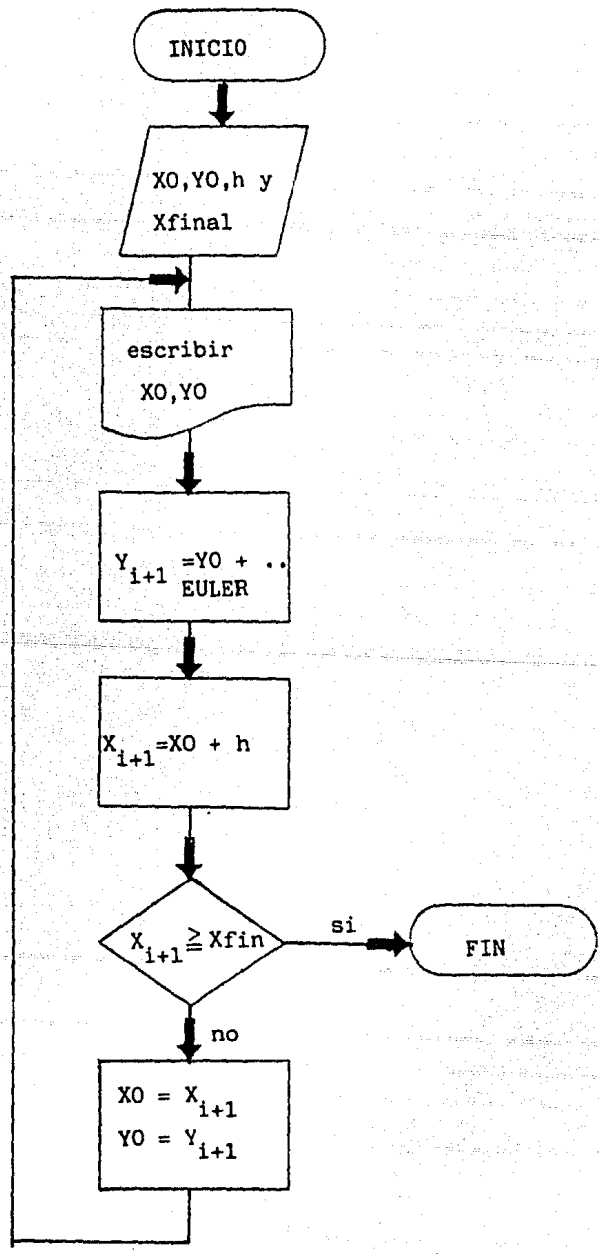
$$Y_i = Y_{i+1}$$

Regresar al inciso 3.

El diagrama de flujo para el método de Eüler se muestra en la figura Nº 17,las subrutinas correspondientes en las siguientes hojas.

FIGURA Nº 17

DIAGRAMA DE FLUJO PARA EL METODO DE EULER



```
    0.      0 C
    1.      0 C *** METODO DE EULER ***
    2.      0 C
    3.      0 C 1) H=TAMANO DE PASO
    4.      0 C 2) XO,YO=CONDICIONES INICIALES
    5.      0 C 3) XFINAL=VALOR FINAL
    6.      0 C
    7.      0       SUBROUTINE EULER(XO,YO,XFINAL,H)
    8.      0       N=(XFINAL-XO)/H
    9.     15       WRITE(6,75)
   10.     35    75 FORMAT(4X,'NO.',9X,'X',9X,'Y')
   11.     35       DO 550 I=1,N
   12.     43       WRITE(6,150)I,XO,YO
   13.     79   150 FORMAT(4X,I2,2(5X,F10.4))
   14.     79       YCALC=YO+H*FUNC(XO,YO)
   15.     97       XO=XO+H
   16.    107       YO=YCALC
   17.    114   550 CONTINUE
   18.    126       RETURN
   19.    128       END

EULER   SUBROUTINE              2
FUNC    REAL       FUNCTION     3,FWD
H       REAL           1*
I       INTEGER        7
N       INTEGER        5
XO      REAL           4*
XFINAL  REAL           2*
YO      REAL           3*
YCALC   REAL           8
```

```
(*                              *)
(* METODO DE EULER              *)
(* LENGUAJE:PASCAL              *)
(*                              *)
PROCEDURE EULER(VAR H,XFINAL,XO,YO:REAL);
 VAR
   N,I:INTEGER;
   YCALC:REAL;
    BEGIN
     N:=(XFINAL-XO)/H;
     WRITELN(' ':3,'NO.',' ':6,'X',' ':6,'Y');
     FOR I:=1 TO N DO
      BEGIN
       WRITELN(' ':3,I:2,' ':6,XO:10:4,' ':6,YO:10:4);
        YCALC:=YO+H*FUNC(XO,YO);
        XO:=XO+H;
        YO:=YCALC;
      END;
   END;(*FIN*)
```

```
 LIST

5000  REM
5010  REM  METODO DE EULER
5020  REM  LENGUAJE:BASIC
5030  REM
5040 N = (XF - XO) / H
5050  PRINT "NO.", SPC( 4),"X", SPC( 4),"Y"
5060  PRINT "---", SPC( 4),"-", SPC( 4),"-"
5070  PRINT
5080  FOR I = 1 TO N
5090  PRINT I, SPC( 4),XO, SPC( 4),YO
5100 YC = YO + H *  FN A(XO,YO)
5110 XO = XO + H
5120 YO = YC
5130  NEXT I
5140  RETURN

]
```

B) Método de Runge-Kutta.

Los métodos de Runge-Kutta fueron los primeros que se emplearon en la solución numérica de ecuaciones diferenciales,la ventaja principal de estos métodos radica en el hecho de que son fáciles de programar,mientras que su desventaja principal estriba en que la derivada f'(x,y) se debe evaluar para varios valores en cada paso de la solución,lo que produce un aumento en el tiempo de computación,estas evaluaciones pueden ser dos,tres ó cuatro,en este capítulo solamente se expone el método de Runge-Kutta de cuarto orden.

Algoritmo

1.- Conocer las condiciones iniciales X0,Y0.

2.- Fijar el tamaño de paso h y X_{final} .

3.- Calcular las evaluaciones de la f'(X,Y) como:

$$K0 = h*f'(X_i, Y_i)$$

$$K1 = h*f'(X_i + h/2, Y_i + K0/2)$$

$$K2 = h*f'(X_i + h/2, Y_i + K1/2)$$

$$K3 = h*f'(X_i + h, Y_i + K2)$$

4.- Calcular el nuevo valor de Y como Y_{i+1} con la siguiente expresión:

$$Y_{i+1} = Y_i + 1/6*(K0 + 2*K1 + 2*K2 + K3)$$

5.- incrementar la variable independiente X como:

$$X_{i+1} = X_i + h$$

6.- Detener el proceso cuando se cumpla lo siguiente :

$$X_{i+1} \geq X_{final}$$

7.- Asignar :

$$X_i = X_{i+1}$$

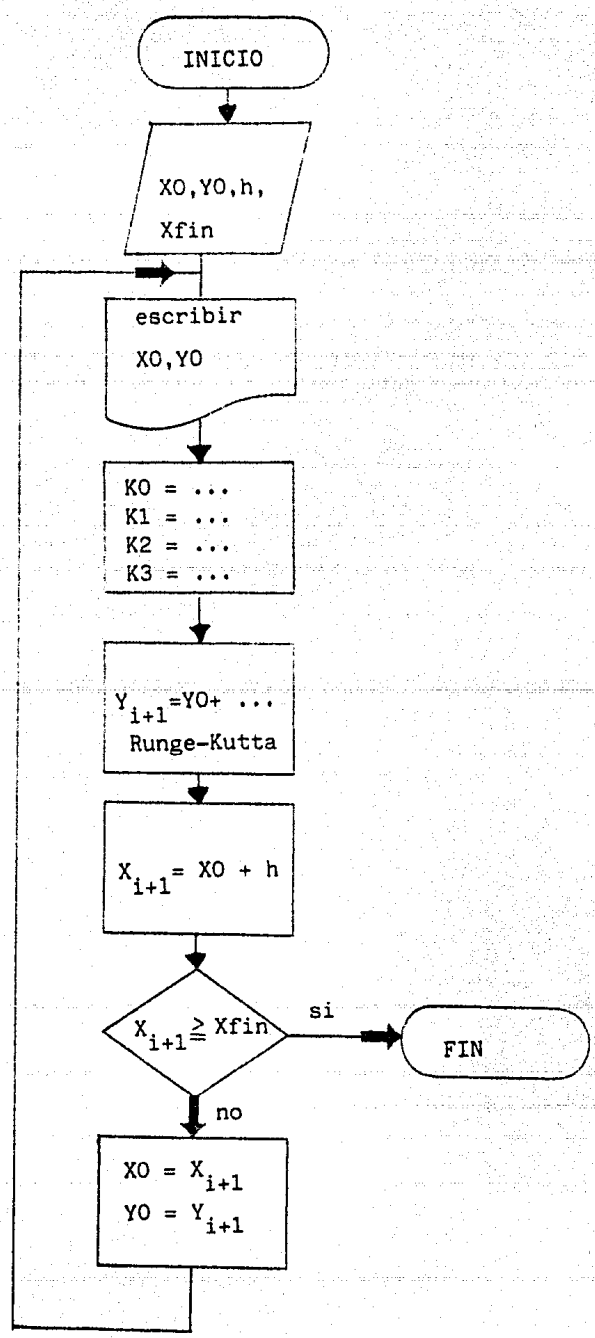
$$Y_i = Y_{i+1}$$

regresar al inciso (3).

El diagrama de flujo se muestra en la figura Nº 18, las subrutinas correspondientes el las siguientes hojas.

FIGURA Nº 18

DIAGRAMA DE FLUJO PARA EL METODO DE RUNGE-KUTTA



```
FORTRAN Compiler II.1  [1.1]
    0.      0 C
    1.      0 C *** METODO DE RUNGE-KUTTA 4 ORDEN ***
    2.      0 C
    3.      0 C 1) H=TAMANO DE PASO
    4.      0 C 2) X0,Y0=CONDICIONES INICIALES
    5.      0 C 3) XFINAL=VALOR FINAL
    6.      0 C
    7.      0       SUBROUTINE KUTTA(X0,Y0,XFINAL,H)
    8.      0       REAL K
    9.      0       N=(XFINAL-X0)/H
   10.     15       WRITE(6,90)
   11.     35    90 FORMAT(4X,'NO.',9X,'X',9X,'Y')
   12.     35       DO 15 I=1,N
   13.     43       WRITE(6,123)I,X0,Y0
   14.     79   123 FORMAT(4X,I2,2(5X,F10.4))
   15.     79       K0=H*FUNC(X0,Y0)
   16.     93       K1=H*FUNC(X0+H/2,Y0+K0/2)
   17.    135       K2=H*FUNC(X0+H/2,Y0+K1/2)
   18.    177       K3=H*FUNC(X0+H,Y0+K2)
   19.    214       YCALC=Y0+1./6.*(K0+2.0*K1+2.0*K2+K3)
   20.    264       X0=X0+H
   21.    274       Y0=YCALC
   22.    281    15 CONTINUE
   23.    293       RETURN
   24.    295       END

FUNC    REAL       FUNCTION      3,FWD
H       REAL           1*
I       INTEGER        7
K       REAL       *****
K0      INTEGER        8
K1      INTEGER        9
K2      INTEGER       14
K3      INTEGER       19
KUTTA   SUBROUTINE                 2
N       INTEGER        5
X0      REAL           4*
XFINAL  REAL           2*
Y0      REAL           3*
YCALC   REAL          24
```

```
(*                              *)
(* METODO DE RUNGE-KUTTA        *)
(* LENGUAJE:PASCAL              *)
(*                              *)
PROCEDURE KUTTA(VAR H,XFINAL,X0,Y0);
 VAR
   K0,K1,K2,K3,YCALC:REAL;
   N,I:INTEGER;
    BEGIN
     N:=(XFINAL-X0)/H;
     WRITELN(' ':3,'NO.',' ':4,'X',' ':4,'Y');
     FOR I:=1 TO N DO
      BEGIN
       WRITELN(' ':3,I:2,' ':3,X0:10:4,' ':4,Y0:10:4);
       K0:=H*FUNC(X0,Y0);
       K1:=H*FUNC(X0+H/2,Y0+K0/2);
       K2:=H*FUNC(X0+H/2,Y0+K1/2);
       K3:=H*FUNC(X0+H,Y0+K2);
       YCALC:=Y0+1/6*(K0+2.0*K1+2.0*K2+K3);
       X0:=X0+H;
       Y0:=YCALC;
      END;
    END;(*FIN*)
```

```
 LIST

5000  REM
5010  REM  METODO DE RUNGE-KUTTA
5020  REM  LENGUAJE:BASIC
5030  REM
5040 N = (XF - XO) / H
5050  PRINT "NO.", SPC( 4),"X", SPC( 4),"Y"
5060  PRINT "---", SPC( 4),"-", SPC( 4),"-"
5070  FOR I = 1 TO N
5080  PRINT I, SPC( 4),XO, SPC( 4),YO
5090 KO = H *  FN A(XO,YO)
5100 K1 = H *  FN A(XO + H / 2,YO + KO / 2)
5110 K2 = H *  FN A(XO + H / 2,YO + K1 / 2)
5120 K3 = H *  FN A(XO + H,YO + K2)
5130 YC = YO + 1 / 6 * (KO + 2 * K1 + 2 * K2 + K3)
5140 XO = XO + H
5150 YO = YC
5160  NEXT I
5170  RETURN

]
```

C) PROBLEMAS DE APLICACION

Problema Nº 1

Bajo determinadas condiciones,el azúcar en agua se transforma en dextrosa a una velocidad proporcional a la cantidad de azúcar sin transformar, sabiendo que para t = 0 minutos, la cantidad de azúcar inicial es de 75 gramos y que al cabo de t = 30 minutos, se han transformado 8 gramos. Hallar la cantidad de dextrosa al cabo de 90 minutos, donde q = es la cantidad de dextrosa transformada en t minutos.

DATOS:

AZUCAR Co = 75.0 gramos

DEXTROSA q = 8.0 gramos

TIEMPO To = 30.0 minutos

CTE. K = 0.0038 $minutos^{-1}$

La ecuación que relaciona estos datos es la siguiente:

$$dq/dt = K(Co - q)$$

antes de aplicar los métods anteriores fijamos el tamaño de paso h como h = 5.0 minutos, los resultados de este problema se aprecian en las tablas Nº 14,15 y 16.

TABLA Nº 14

LENGUAJE PASCAL

EULER	RUNGE-KUTTA	No.	Tiempo	EULER	RUNGE-KUTTA
1.09*	1.65*	1	30	8.0	8.0
		2	35	9.2730	9.2730
		3	40	10.5218	10.4982
		4	45	11.7468	11.7121
		5	50	12.9487	12.9033
		6	55	14.1276	14.0720
		7	60	15.2842	15.2187
		8	65	16.4188	16.3438
		9	70	17.5318	17.4477
		10	75	18.6237	18.5309
		11	80	19.6949	19.5937
		12	85	20.7457	20.6365
		13	90	21.7765	21.6596

*Tiempo medido en segundos.

.*Promedio de 15 ejecuciones.

TABLA Nº 15

LENGUAJE FORTRAN

EULER	RUNGE-KUTTA	No.	Tiempo	EULER	RUNGE-KUTTA
1.17*	1.84*	1	30	8.0	8.0
		2	35	9.2730	9.2730
		3	40	10.5218	10.4982
		4	45	11.7468	11.7121
		5	50	12.9487	12.9033
		6	55	14.1276	14.0720
		7	60	15.2842	15.2187
		8	65	16.4188	16.3438
		9	70	17.5318	17.4477
		10	75	18.6237	18.5309
		11	80	19.6949	19.5937
		12	85	20.7457	20.6365
		13	90	21.7765	21.6596

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

TABLA Nº 16

LENGUAJE BASIC

EULER	RUNGE-KUTTA	No.	Tiempo	EULER	RUNGE-KUTTA
1.32*	2.22*	1	30	8.0	8.0
		2	35	9.2730	9.2609
		3	40	10.5218	10.4982
		4	45	11.7468	11.7121
		5	50	12.9487	12.9033
		6	55	14.1276	14.0720
		7	60	15.2842	15.2187
		8	65	16.4188	16.3438
		9	70	17.5318	17.4477
		10	75	18.6237	18.5309
		11	80	19.6949	19.5937
		12	85	20.7457	20.6365
		13	90	21.7765	21.6596

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

Problema Nº 2

Un tanque que contiene 80 lb de sal forma una solución salina de 250 galones.

Otra solución salina que contiene 25 lb de sal por galón, se vierte en el tanque a razón de 30 galones/minuto, mientras que simultáneamente, la solución ya mezclada sale del tanque a razón de 45 galones/minuto. Encontrar la cantidad de sal que hay en el tanque en el tiempo t = 50 minutos.

DATOS :

Vo = 250 galones iniciales.
f = 45 galones/minuto de salida.
e = 30 galones/minuto de entrada.
b = 25 lb de sal por galón.
To = 0 minutos.
Q = 80 lb de sal en el tanque.
Tf = 50 minutos.

La ecuación que relaciona al sistema es la siguiente:

$$dQ/dt = b*e - f*Q/(Vo + (e - f)*t)$$

aplicando los métodos anteriores y fijando un tamaño de paso h = 5.0 minutos, los resultados de este problema se aprecian en las tablas Nº 17,18 y 19.

TABLA Nº 17

LENGUAJE PASCAL

EULER	RUNGE-KUTTA	No.	Tiempo	EULER	RUNGE-KUTTA
1.23*	2.31*	1	0	80.0	80.0
		2	5	125.8151	117.0228
		3	10	153.9852	144.1036
		4	15	174.7381	164.7786
		5	20	214.1988	203.1958
		6	25	234.2773	223.3519
		7	30	251.6686	240.8925
		8	35	267.5165	256.9065
		9	40	282.2544	271.8119
		10	45	296.1209	285.8419
		11	50	309.2711	299.1491

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

TABLA Nº 18

LENGUAJE FORTRAN

EULER	RUNGE-KUTTA	No.	Tiempo	EULER	RUNGE-KUTTA
1.58*	2.62*	1	0	80.0	80.0
		2	5	125.8161	117.0228
		3	10	153.9852	144.1036
		4	15	174.7381	164.7786
		5	20	214.1988	203.1958
		6	25	234.2773	223.3519
		7	30	251.6686	240.8925
		8	35	267.5165	256.9065
		9	40	282.2544	271.8119
		10	45	296.1209	285.8419
		11	50	309.2711	299.1491

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

TABLA Nº 19

LENGUAJE BASIC

EULER	RUNGE-KUTTA	No.	Tiempo	EULER	RUNGE-KUTTA
1.46*	2.97*	1	0	80.0	80.0
		2	5	125.8161	117.0228
		3	10	153.9852	144.1036
		4	15	174.7381	164.7786
		5	20	214.1988	203.1958
		6	25	234.2773	223.3519
		7	30	251.6686	240.8925
		8	35	267.5165	256.9065
		9	40	282.2544	271.8119
		10	45	296.1209	285.8419
		11	50	309.2711	299.1491

*Tiempo medido en segundos.

*Promedio de 15 ejecuciones.

D) ANALISIS DE RESULTADOS

Podemos observar en las tablas anteriores que no hay mucha diferencia en los tiempos de ejecución en los tres lenguajes ya que en promedio son solo centésimas de segundo,para la solución de problemas empleando los métodos de Eüler y Runge-Kutta, aunque,la diferencia aumenta al hacerse más compleja la ecuación diferencial.

Finalmente podemos decir que el método más exacto en la resolución de los dos problemas planteados es el de Runge-Kutta.

C A P I T U L O V I I

VII.- CONCLUSIONES

Con el presente trabajo podemos concluir que en el manejo y resolución de problemas a traves de microcomputadoras los principales factores que se deben considerar son:

1) El tiempo de proceso de cada método,ya que cada subrutina puede ser usada varias veces en un programa principal,incrementando considerablemente el tiempo global de ejecución.

2) La elección del método,ya que si se elige un método lento, el tiempo de proceso se verá incrementado,en general,el criterio de elección del método dependerá en gran parte de las característica del problema por resolver.

3) La capacidad de almacenamiento que se tenga en la microcomputadora empleada ya que es importante saber que en promedio,las subrutinas sin líneas de comentarios,usan 850 caracteres (bytes) de su memoria principal RAM.

4) El tipo de lenguaje que se dispone en la microcomputadora, ya que generalmente el lenguaje es el BASIC,aunque como se observó en este trabajo el lenguaje que es más rápido en - procesamiento es PASCAL.

Finalmente,podemos concluir también que los objetivos trazados al principio del trabajo se han alcanzado,dejando como base ciertos criterios que permiten la elección de algún método numérico para resolver algún problema característico, además de permitir el acceso a esta nueva tecnología de las microcomputadoras.

C A P I T U L O V I I I

VIII.- BIBLIOGRAFIA

a) Gillet & Carlile," More on Matrix Theory "
The Oil & Gas Journal,April 22,1968, p194.

b) Carlile,R.E. & Gillet," Gauss-Jordan Technique for
Solving Linear Systems ".
The Oil & Gas Journal,July 29,1968,p82.

c) Idem," Gauss-Seidel Technique for Solving Linear
Systems ".
The Oil & Gas Journal,Agosto 19,1968 p87.

d) Southworth & Dedeeuw." Digital Computation and
Numerical Methods ",McGraw-Hill,1965.

e) Hamming R.W.," NUmerical Methods for Scientists and
Engineers ",McGraw-Hill,1962.

f) Apple," Apple Pascal Operating System Reference Manual "
APPLE COMPUTER INC.,1980.

g) Apple," Apple Pascal Language Reference Manual "
APPLE COMPUTER INC. 1980.

h) Apple," Applesoft Basic Programming Reference Manual "
APPLE COMPUTER INC.,1978.

i) Apple," Apple FORTRAN Reference Manual ".
APPLE COMPUTER INC.,1980.

j) Murril & Smith," BASIC PROGRAMMING ".
International Textbook Co.,1971.