

**UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO**

---

**FACULTAD DE CIENCIAS**



**BIBLIOTECA**  
**INSTITUTO DE ECOLOGIA**  
*UNAM*

**DESARROLLO DE METODOS  
MONTECARLO, EN APL**

**T E S I S**

Que para obtener el título de:

**M A T E M A T I C O**

**p r e s e n t a**

**BERNARDO FENIG BINDER**

México, D. F.

1976

A MI PADRE

CUYA SABIDURIA ME HA GUIADO POR LA VIDA

A MI MADRE

CUYA COMPRESION ME HA IMPULSADO SIEMPRE

Deseo hacer patente mi agradecimiento a las siguientes personas e instituciones:

Al Dr. Eugenio Mendoza Villarreal, investigador titular del Instituto de Astronomía, U.N.A.M., gracias a cuya atinada dirección fue posible la realización de esta tesis.

Al Centro Científico de I.B.M. de México S.A. por las facilidades prestadas en los servicios de cómputo referentes a esta tesis.

A los Dres. José Negrete Martínez y Guillermina Yankelevich Nedvedovich, investigadores titulares del Instituto de Investigaciones Biomédicas U.N.A.M., por sus valiosas asesorías.

Al Departamento de Servicios de Cómputo de la Universidad Autónoma Metropolitana Unidad Iztapalapa por haberme ofrecido la oportunidad de usar sus instalaciones de comunicación de datos.

Al Centro de Servicios de Cómputo, U.N.A.M., gracias a cuyas facilidades pudieron probarse algunos programas.

A todos mis compañeros y amigos que me impulsaron y apoyaron.

## INDICE

INTRODUCCION AL <i>APL</i> .....	1
INTRODUCCION AL METODO MONTECARLO.....	21
NUMEROS ALEATORIOS.....	28
INTEGRACION MULTIPLE.....	64
CONCLUSIONES.....	95

## INTRODUCCION AL APL.

El lenguaje tradicional usado por las Matemáticas posee muchas deficiencias que frecuentemente acarrearán problemas, tales como la falta de comunicación entre las personas que se dedican a transmitir y recibir ideas matemáticas.

Esta fue una de las principales razones que motivaron a Kenneth E. Iverson a construir un nuevo lenguaje matemático que viniera a terminar con los grandes defectos que adolecía el lenguaje tradicional.

Mencionaremos algunos de los más grandes inconvenientes del álgebra tradicional:

1.-La ambigüedad. Esto quizás es uno de los más grandes problemas del álgebra tradicional. Por ejemplo, cuando tenemos la expresión

$$c=ab$$

no sabemos si  $ab$  es una variable por sí misma o si es el producto de  $a$  por  $b$ . Por otro lado, con respecto al signo '=', ignoramos si se le asigna a  $c$  el valor del resultado de las operaciones de la derecha, o si simplemente se está estableciendo una hipótesis acerca de si  $c$  es igual a  $ab$ .

2.-La arbitrariedad para definir variables. En el álgebra tradicional se pueden definir casi arbitrariamente variables sin seguir una regla definida. Prueba de ello es la utilización indistinta de letras mayúsculas o minúsculas del alfabeto español, griego, hebreo, etc. Existe cierta anarquía, pues cada persona adopta de hecho su propia notación.

3.-La anarquía en la sintaxis. Debido a causas ajenas a nosotros, se nos ha heredado una sintaxis bastante arbitraria en cuanto al lenguaje matemático. Por ejemplo, si queremos calcular el seno de  $X$ , primero se pone el operador y luego el argumento

$$\text{SEN } X$$

pero sucede al revés con el factorial

X!

y en el caso del valor absoluto de un número, éste tiene un símbolo que lo precede y otro que lo sigue

|X|

4.-Reglas arbitrarias y confusas en lo referente a prioridades de los operadores. Este es un problema que a menudo causa dificultades, sobre todo cuando se trata de expresiones algebraicas muy complicadas.

Todas estas fallas son debidas a que este lenguaje ha surgido según las necesidades que se han planteado en toda la historia de las Matemáticas, y los grandes matemáticos no se han preocupado mucho en corregir estos errores.

*APL* (A Programming Language) es un intento bastante fructífero de terminar con todos los vicios del lenguaje tradicional. Vale la pena aclarar que *APL* no es solamente un lenguaje de programación más, como Algol o Fortran, sino que es una redefinición del lenguaje algebraico que tiene extensiones muy poderosas y útiles en la resolución de problemas de variadas índoles.

*APL* tiene la gran virtud de no exigir como pre-requisito el conocimiento de sistemas computacionales para el principiante. Aún sin el auxilio de una computadora, el *APL* puede substituir eficientemente al lenguaje tradicional en una cátedra de Matemáticas.

Entrando en materia, un número escalar en *APL* se define en cualquiera de las siguientes maneras:

- 1.-Una sucesión no vacía de dígitos (0,1,2,3,4,5,6,7,8,9).
- 2.-Un carácter '-' (léase negativo, que es distinto a la función '-' que se verá posteriormente) seguido de una sucesión no vacía de dígitos.
- 3.-Una sucesión de dígitos seguida de un '.' (léase punto) seguido de una sucesión de dígitos (al menos un dígito a la derecha o izquierda del punto).
- 4.-Un carácter '-' seguido de una sucesión de dígitos, seguido de un punto, seguido de una sucesión de dígitos (al menos un dígito a la derecha o izquier-

da del punto).

5.-Cualquiera de los casos anteriores seguido de una 'E',seguido de cualquiera de los casos anteriores.

Ejemplos válidos de números escalares son:

$^{-}8$

3.1416

$1E^{-}10$

.5

0.0

Una variable en APL es cualquiera de los siguientes caracteres: A B C D E F  
G H I J K L M N O P Q R S T U V W X Y Z  $\Delta$   $\underline{A}$   $\underline{B}$   $\underline{C}$   $\underline{D}$   $\underline{E}$   $\underline{F}$   $\underline{G}$   $\underline{H}$   $\underline{I}$   $\underline{J}$   $\underline{K}$   $\underline{L}$   $\underline{M}$   $\underline{N}$   $\underline{O}$   $\underline{P}$   $\underline{Q}$   $\underline{R}$   
 $\underline{S}$   $\underline{T}$   $\underline{U}$   $\underline{V}$   $\underline{W}$   $\underline{X}$   $\underline{Y}$   $\underline{Z}$   $\underline{\Delta}$ ,seguido de una sucesión arbitraria de estos caracteres o dígitos(nótese que una variable no puede comenzar con un dígito).

Una vez hecho esto,podemos empezar a estudiar las funciones primitivas,que son aquellas que nos proporciona el sistema APL.Adiferencia del Algebra tradicional,el APL no distingue prioridad entre sus operaciones.APL ejecutará sus funciones siguiendo un orden estricto de derecha a izquierda.La única manera de modificar este orden es la introducción de los paréntesis '(' y ')', que siguen las mismas reglas que el álgebra tradicional.Siempre se ejecutará primero una expresión encerrada entre un par de paréntesis.De no cumplirse estrictamente las reglas sintácticas de paréntesis,entonces se producirá un error de sintaxis.Las reglas son las siguientes:

- 1.-Al recorrerse una expresión de izquierda a derecha,el número de paréntesis '(' siempre será menor o igual que ')'
- 2.-Al terminar la expresión,el número total de ambas clases de paréntesis debe coincidir.

Las funciones primitivas se dividen en dos grupos principales:monádicas y diádicas.

Funciones monádicas son aquellas que sólo tienen un argumento a su derecha. las funciones diádicas tienen un argumento a su derecha y otro a su izquierda.

Empezemos por definir las funciones más elementales, y que son las llamadas funciones primitivas escalares:

'+'(identidad) en su forma monádica deja a su argumento invariable.

'+'(adición) en su forma diádica suma el argumento de la derecha con el de la izquierda.

$$+7.5$$

7.5

$$3+8.5$$

11.5

$$++5+5$$

10

'-'(simétrico) en su forma monádica nos entrega el inverso aditivo del número en cuestión. Aquí hacemos un paréntesis para aclarar que también existe el símbolo '-'(negativo) que no es una función sino que es parte de un número y con eso se hará notar que el número es menor o igual que cero.

En su forma diádica la función '-'(sustracción) nos entrega la resta teniendo como minuendo el primer argumento y como sustraendo el segundo.

$$6-3.5$$

2.5

$$-8$$

8

$$-8-5$$

13

'x'(signo) en su forma monádica nos entrega sólo tres valores, a saber:



1, si el argumento es mayor que cero.

0, si el argumento es cero.

-1, si el argumento es menor que cero.

'\*' (multiplicación) en su forma diádica multiplicará el argumento izquierdo por el derecho.

$$\begin{array}{r}
 3 \times 5 \\
 15 \\
 \times 7 \\
 \hline
 21 \\
 \times 3 \times 5 \\
 1
 \end{array}$$

'/' (recíproco) en su forma monádica nos entrega el inverso multiplicativo del argumento. Nótese que es un error intentar esta función con el argumento cero.

'/' (división) efectúa la división del argumento izquierdo (dividendo) entre el argumento derecho (divisor) en su forma diádica. También es un error si el argumento derecho tiene valor cero.

$$\begin{array}{r}
 \div 5 \\
 0.2 \\
 \div 5 \\
 5 \\
 2 \div 4 \\
 0.5
 \end{array}$$

'|' (piso) en su forma monádica nos entrega el máximo entero menor o igual que el argumento.

'|' (mínimo) en su modalidad diádica nos entrega el menor número entre su argumento derecho e izquierdo.

'|'(techo) monádica, nos proporciona como resultado el mínimo entero mayor o igual que el argumento.

'|'(máximo) diádica da como valor el mayor entre ambos argumentos.

[5.7

6

[5.7

5

[4.4

5

[3.99

3

[4

4

[3.7[3.5

4

'\*(exponencial) monádica nos entrega el número e(2.817281728....) elevado al argumento especificado.

'\*(potenciación) da como valor el argumento izquierdo elevado a la potencia especificada por el argumento derecho.

'\*(logaritmo natural) monádica es el logaritmo natural del argumento (que tiene que ser mayor que cero).

'\*(logaritmo) en su versión diádica nos da el logaritmo con base en el argumento izquierdo del argumento derecho (ambos tienen que ser mayores que cero).

'|'(valor absoluto) monádica es igual al argumento si este es mayor o igual que cero y el simétrico del argumento si éste es menor que cero.

'|'(residuo) es en su forma diádica definida como sigue:

$$A|B \leftrightarrow B - (|A|) \times |B| |A| \text{ si } A \neq 0$$

$$B \text{ si } A=0 \text{ y } B>0$$

y nos da error en cualquier otro caso.

5|2  
 2  
 2|5  
 1  
 |<sup>-</sup>5.7  
 5.7  
<sup>-</sup>3|<sup>-</sup>5  
<sup>-</sup>2

!!(factorial) !A en su forma monádica se define en general como la función Gamma de A+1,y si es entero no negativo, en particular, como el factorial del argumento A.

!!(coeficiente binomial) se define como

$$A!B \leftarrow \frac{A!B!}{A! \times B!}$$

!o'(pi veces 3.14159...) en su forma monádica nos da el número PI multiplicado por su argumento.

En su forma diádica AOB nos entrega lo siguiente:

A	resultado
0	(1-B*2)*0.5
1	seno de B
<sup>-</sup> 1	ángulo cuyo seno es B
2	coseno de B
<sup>-</sup> 2	ángulo cuyo coseno es B
3	tangente de B
<sup>-</sup> 3	ángulo cuya tangente es B
4	(1+B*2)*0.5
<sup>-</sup> 4	( <sup>-</sup> 1+B*2)*0.5
5	seno hiperbólico de B

A	resultado
$\bar{5}$	ángulo cuyo seno hiperbólico es B
6	coseno hiperbólico de B
$\bar{6}$	ángulo cuyo coseno hiperbólico es B
7	tangente hiperbólica de B
$\bar{7}$	ángulo cuya tangente hiperbólica es B.

' $\bar{\sim}$ ' (negación) sólo tiene versión monádica y únicamente puede tener dos argumentos: 0 ó 1. Si su argumento es 0 nos entrega 1 y si es 1 nos entrega 0.

Las funciones ' $\wedge$ ' (conjunción), ' $\vee$ ' (disyunción), ' $\bar{\wedge}$ ' (negación de la conjunción), ' $\bar{\vee}$ ' (negación de la disyunción) sólo pueden tener en ambos argumentos 0 ó 1 y todas ellas son diádicas. Los resultados están dados por la siguiente tabla:

A	B	$A \wedge B$	$A \vee B$	$\bar{A \wedge B}$	$\bar{A \vee B}$
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	1	0	0

Las funciones de relación '<' (menor), ' $\leq$ ' (menor o igual), '=' (igual), ' $\geq$ ' (mayor o igual), '>' (mayor), ' $\neq$ ' (distinto de) son todas ellas diádicas y sólo pueden entregar dos tipos de valores:

- 0 si la relación es falsa,
- 1 si la relación es verdadera.

$\bar{7} \neq 8$

1

$\bar{7} = 8$

0

$3 \leq 5$

1

<3>5

1

La función '?' (selección al azar) será de gran utilidad posteriormente en este trabajo. En su forma monádica nos genera un entero al azar entre 1 y el entero positivo (forzosamente) que se le da como argumento.

Hasta ahora se han visto una serie de funciones que aparentemente están definidas sólo para escalares. Una de las grandes virtudes de APL es que estas funciones son manejables con otro tipo de estructuras, a saber:

Vectores, matrices e hipermatrices.

Las funciones monádicas funcionan con cualquier tipo de estas estructuras, y lo que hacen es que calculan la función de cada elemento de la estructura, generando una estructura de las mismas características en cuanto a dimensiones.

```

      × 0 1 2.5 3 4
0 1 1 1 1
      P
3.7 4.1 5.7 1.2
2 6 3.8 0
      LP
3 4 5 2
2 6 3 0

```

Las funciones diádicas operan bajo las siguientes restricciones:

- 1.-Un escalar y cualquier estructura como argumentos.
- 2.-Dos estructuras de la misma dimensión como argumentos.

En el primer caso se aplica la función del escalar con cada uno de los elementos de la estructura dando como resultado una estructura de la misma dimensión. En el segundo caso el resultado se obtiene operando los elementos coincidentes en cuanto a su posición en ambos arreglos.

2|5 3 2 1

1 1 0 1

A

4 7 6

5 6 4

4 0 4

B

6 3 2

4 3 1

4 7 9

A>B

0 0 1

0 1 1

0 0 0

1 0 0 0 0 1 0 0 1 1 0 0

1 0 1 1 0 1

Para representar un vector en *APL* bastará poner sus coordenadas separadas por cuando menos un espacio en blanco entre sí

5 7 6 6 0 2

será un vector de dimensión 6.

La representación de vectores y matrices será indirecta, y su generación se verá posteriormente.

La función '*i*' (generador de índices) es de gran importancia y sólo acepta como argumento en su forma monádica un entero no negativo. El resultado, si se aplica la función '*i*' al escalar *N* es un vector de *N* dimensiones con los primeros *N* enteros (sin incluir al cero). Si se le aplica al escalar 0, el resultado es el vector vacío.

16  
 1 2 3 4 5 6  
 10

Otra función importante es la función ' $\rho$ ' (dimensión) que en su forma monádica nos entrega un vector de dimensión  $N$  (donde  $N$  es el espacio de la estructura de la cual se trata). Por ejemplo, vector, dimensión 1, matriz, dimensión 2, hipermatriz en el espacio tridimensional, dimensión 3, etc. Los valores de cada coordenada son las dimensiones de renglones, columnas, etc.

$\rho_{t5}$   
 5  
 4  
 3 4 1  
 0 2 3

$\rho_A$   
 2 3  
 B  
 2 5 7  
 3 4 2  
 2 1 0  
 2 22 4

3 4 7  
 1 5 6

1 1 1  
 1 1 1  
 $\rho_B$

4 2 3

En su forma diádica, la función 'p' (reestructuración), es la generadora de cualquier tipo de estructura que maneja APL. Su argumento izquierdo debe de ser un vector con elementos enteros no negativos, cuya dimensión vendrá a ser el espacio del arreglo (1, vector, 2, matriz, 3, hipermatriz en el espacio tridimensional, etc.). El argumento de la derecha puede ser cualquier estructura algebraica y la va a recorrer principiando con la última coordenada del espacio (en el caso de matrices, por ejemplo, primero columnas y luego renglones).

El resultado de esta operación va a ser llenado también empezando por la última coordenada del espacio y si no tiene suficientes elementos, regresará otra vez al principio para seguir llenando hasta concluir. Puede también darse el caso de que haya sobrantes en los elementos de la estructura del argumento derecho. los ejemplos a continuación ilustrarán mejor esta situación.

```

      2p1+12
2 3
      2 3p14
1 2 3
4 1 2
      2 3 4p2 3p14
1 2 3 4
1 2 3 4
1 2 3 4

1 2 3 4
1 2 3 4
1 2 3 4

```

Antes de seguir adelante, debemos definir una función vital para APL, ya que



a través de ella podemos almacenar valores que después podrían usarse. Nos referimos a la función '+' (asignación) que solamente puede tener forma diádica. El argumento de la izquierda debe de ser siempre una variable y el de la derecha puede ser una variable o cualquier expresión de APL válida. Si a la derecha se usa cualquier variable, ésta debe de estar definida anteriormente.

A←2 3p13

B←A|5

C←0.5×C+A|B

En álgebra tradicional juegan un papel muy importante los símbolos  $\Sigma$  y  $\Pi$ . En APL es muy fácil manejar este tipo de operaciones que no sólo se restringen a la suma y la multiplicación, sino a cualquier función primitiva escalar diádica. El nombre de la función es reducción y se expresa de la siguiente manera:

f/A

donde f es la función primitiva escalar diádica que se va a aplicar al argumento A. Si A es el vacío, nos entregará el elemento neutro de la función (si existe). A puede ser cualquier estructura numérica. Si es un escalar, lo deja invariante; si se trata de un vector, aplica la función a cada par de elementos o resultados parciales empezando por la derecha

+/16

21

-/15

3

En el caso de matrices se aplica a las columnas, dando un vector

×/4 5p1+14

240 360 480 600

Siempre el resultado será una estructura en una dimensión del espacio más baja, y siempre se aplicará a la última coordenada del argumento.

Si en el caso de matrices o hipermatrices se quiere que la operación se aplique a la primera coordenada del espacio, en vez del símbolo '/' se pondrá el símbolo '!'.

$\times/4 \ 5p1+i4$   
120 120 120 120 120

En el caso de hipermatrices, si se quiere que esto se haga en coordenadas interiores, entonces se tiene que especificar el número de coordenada tomando en cuenta que la primera coordenada es la más interna, o sea que si

$A \leftarrow 2 \ 4 \ 6p2 \ 3$   
entonces  $L/A$  es equivalente a  $L/[1]A$  y  $L/A$  sería lo mismo que  $L/[3]A$ , pero en este caso si tenemos

$A \leftarrow 2 \ 4 \ 6p110$

$A$   
1 2 3 4 5 6  
7 8 9 10 1 2  
3 4 5 6 7 8  
9 10 1 2 3 4

5 6 7 8 9 10  
1 2 3 4 5 6  
7 8 9 10 1 2  
3 4 5 6 7 8

$L/A$   
1 1 3 1  
5 1 1 3

$L/[2]A$   
1 2 1 2 1 2  
1 2 3 4 1 2

L7A

1 2 3 4 5 6

1 2 3 4 1 2

3 4 5 6 1 2

3 4 1 2 3 4

Nótese en el vector del espacio de la matriz que se suprime la coordenada a través de la cual se hizo la operación.

Cuando se desea referirse a ciertas coordenadas de un vector se usan los corchetes '[' y ']'. La forma de hacer esto es poner el nombre del vector, abrir los corchetes y poner el vector de coordenadas que deseamos extraer. Inmediatamente después se cierran los corchetes. Si

A←2+15

A[1 3 5]

3 5 7

A[2]

4

Si ya nos referimos a matrices o hipermatrices es necesario separar las coordenadas de las distintas dimensiones por el carácter ';'. Siempre deberá haber un ';' menos que el espacio del arreglo. Si el carácter ';' va precedido de nada o de otro ';' se referirá a toda la dimensión. Siempre a la izquierda irán las dimensiones más exteriores. Tomemos por ejemplo

A←2 3 4 5 p17

A

1 2 3 4 5

6 7 1 2 3

4 5 6 7 1

2 3 4 5 6

7 1 2 3 4

5 6 7 1 2

3 4 5 6 7

1 2 3 4 5

6 7 1 2 3

4 5 6 7 1

2 3 4 5 6

7 1 2 3 4

5 6 7 1 2

3 4 5 6 7

1 2 3 4 5

6 7 1 2 3

4 5 6 7 1

2 3 4 5 6

7 1 2 3 4

5 6 7 1 2

3 4 5 6 7

1 2 3 4 5

6 7 1 2 3

4 5 6 7 1

A[2;3;4;5]

A[2;2;2;2]

3

A[1;3;;]

6 7 1 2 3

4 5 6 7 1

2 3 4 5 6

7 1 2 3 4

A[2;1;3;]

1 2 3 4 5

A[;;;2]

2 7 5 3

1 6 4 2

7 5 3 1

6 4 2 7

5 3 1 6

4 2 7 5

Estamos ahora listos para definir la función 'i'(posición) en su forma diádica

$V_i S$

El argumento de la izquierda es un vector y el de la derecha es cualquier estructura; nos entrega como valor la coordenada de la primera aparición (de izquierda a derecha) de  $S$  en  $V$ . Si no aparece ni una sola vez entonces tendrá como valor

$(i1)+pV$

La estructura generada será del mismo tipo del argumento a la derecha (que no debe de ser un escalar).

$S+2 \ 3p14$

$S$

1 2 3

4 1 2

$(13)1S$

1 2 3

4 1 2

Una función diádica de gran importancia para este trabajo es la función '2' (distribución al azar) cuyos argumentos son escalares enteros positivos

$A \neq B$

nos entrega un vector de dimensión  $A$  cuyos elementos están escogidos al azar del vector  $1B$ . Nunca se repetirán elementos en el vector resultado, por lo cual  $A$  debe de ser menor o igual que  $B$ .

$5 \neq 6$

6 2 1 4 3

La función ',' (desarrollo) en su forma monádica convierte su argumento en un vector

$,2 \ 3p6 \ 0 \ 1 \ 4$

6 0 1 4 6 0

En su forma diádica ',' (concatenación) enlaza dos estructuras en una. Si se trata de vectores, lo hace sin condiciones en cuanto a la magnitud. En el caso de espacios superiores, es necesario que coincidan en todas sus dimensiones menos en la más interna.

$2,6p0 \ 1$

2 0 1 0 1 0 1

$(3 \ 2p14), \ 3 \ 4p \bar{1} \ 4 \ 7 \ 2 \ 1$

1 2  $\bar{1}$  4 7 2

3 4 1 1 4 7

1 2 2 1 1 4

Uno de los grandes poderes de APL es que el mismo usuario puede definir funciones propias. Esto se hace mediante el símbolo '∇' (definición de función) que siempre debe anteceder y seguir a cualquier función que se defina. La función puede constar de varios pasos, en cuyo caso se numerarán progresivamente. También puede ser sin argumentos, monádica o diádica.

∇Z←MINIMO A

[1] Z←L/A

[2] ∇

Aquí se trataría de una función monádica que tiene como argumento A; A debe de ser cualquier estructura numérica válida de APL. Ahora sólo basta con llamarla

MINIMO 16

1

MINIMO 676

1

En una función puede haber variables locales, que sólo se usan dentro de la función y después desaparecen al haberse ejecutado ésta. Estas variables se ponen a la derecha de la definición o del argumento derecho, según el caso, se paradas por ';'. .

∇Z←MAXMIN A;B;C

[1] B←L/A

[2] C←I/A

[3] B

[4] C

[5] ∇

Es posible transferir el control de una parte a otra de la función mediante el símbolo ' $\rightarrow$ ' que evalúa cualquier expresión a su derecha y si el paso obtenido existe, se va a él, de otra manera se sale de la función.

$VZ \leftarrow A \text{ MCD } B; C$

[1]  $\rightarrow 2+4 \times 0 = B | A$

[2]  $C \leftarrow A$

[3]  $A \leftarrow B$

[4]  $B \leftarrow B | C$

[5]  $\rightarrow 1$

[6]  $Z \leftarrow B$

[7]  $\nabla$

La función *MCD* calcula el máximo común divisor de dos enteros siguiendo el algoritmo de Euclides.

Estas son sólo algunas de las características de *APL* y a medida que se vaya desarrollando el trabajo se harán nuevas definiciones según sean requeridas.



## INTRODUCCION AL METODO MONTECARLO

La mayoría de la gente está acostumbrada a que en el momento en que surgen problemas referentes a análisis numérico, estos son atacados con los métodos tradicionales conocidos. Todos ellos se refieren a procesos determinísticos, pero llega un momento en que el problema es tan complicado, que estos métodos se vuelven ineficientes e inclusive impotentes para dar la solución.

Como una alternativa, en los años 40 en los Estados Unidos de Norteamérica se empezó a desarrollar un nuevo método que recibió el nombre de Montecarlo, quizás haciendo alusión a los famosos casinos que se localizan en Europa.

Peró esta idea no era nueva ni mucho menos: el primer caso que registra la historia de las Matemáticas en que se utilizan métodos estocásticos para hacer cálculos matemáticos es en el año 1777, en que Buffon descubrió un camino experimental para calcular el número real que establece la proporción entre la longitud del diámetro de una circunferencia y su perímetro. Ese número es conocido comúnmente como  $PI(3.14159\dots)$ .

Los únicos instrumentos que usa son una aguja de longitud  $D[2]$  y una superficie pintada con líneas paralelas a una distancia  $D[1]$  entre ellas.

La aguja es lanzada  $N$  veces en la superficie, y son contadas las ocasiones en que la aguja intersecta a alguna de las rectas. Si llamamos  $ALFA$  al ángulo que forma la aguja con la dirección positiva del sistema de líneas paralelas, sabremos que éste se encuentra restringido a las siguientes condiciones:

$$ALFA \leq 0.5$$

$$ALFA \geq 0.5$$

Por otro lado, si llamamos a la posición del centro de la aguja con el identificador  $A$ , deberá satisfacer las siguientes condiciones:

$$A \geq 0$$

$$A \leq D[1]$$

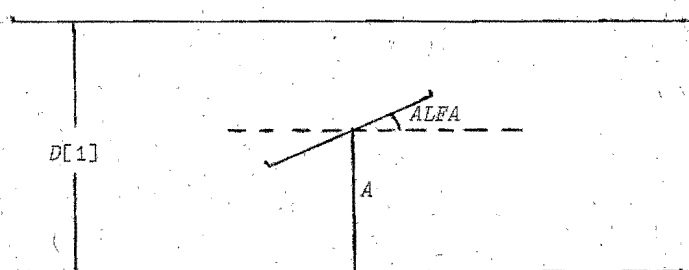


Figura 1.

Si queremos establecer las condiciones bajo las cuales la aguja no intersec-  
ta a ninguna recta, éstas serán simultáneamente:

$$0 < A - 0.5 \times D[2] \times 2 \times \text{ALFA} \quad (1)$$

$$D[1] > A + 0.5 \times D[2] \times 2 \times \text{ALFA} \quad (2)$$

Sabemos que  $A$ , al ser aleatorio, está distribuido uniformemente entre 0 y  $D[1]$  y  $\text{ALFA}$  también lo está entre  $0$  y  $0.5$ .

Una última condición que estableceremos es:

$$D[1] \geq D[2]$$

La probabilidad  $P$  de que las condiciones (1) y (2) se satisfagan simultá-  
neamente puede ser comprendida fácilmente con el siguiente diagrama:

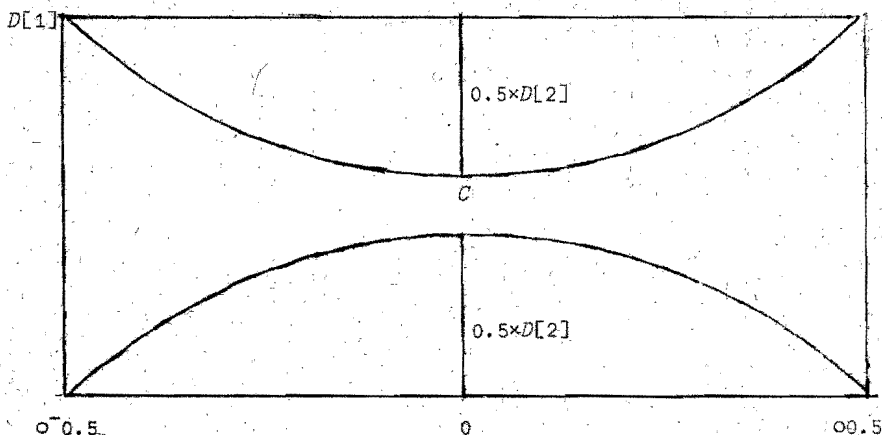


Figura 2.

La región  $C$  es aquella en la cual no se interseca a ninguna línea.

el área del rectángulo es  $OD[1]$  y el área  $C$  es la diferencia de dos integrales:

$$\int_0^{0.5} (D[1] - 0.5 \times D[2] \times 2 \times \text{ALFA}) \text{DALFA} - \int_0^{0.5} (-0.5 \times D[2] \times 2 \times \text{ALFA}) \text{DALFA}$$

lo cual nos da la integral a calcular

$$\int_0^{0.5} (D[1] - D[2] \times 2 \times \text{ALFA}) \text{DALFA}$$

cuyo resultado es

$$C \leftarrow (OD[1]) - 2 \times D[2]$$

La probabilidad será entonces la superficie de la región dividida entre el área del rectángulo, o sea

$$P \leftarrow 1 - (2 \times D[2]) \div OD[1]$$

La probabilidad de que la aguja sí intersekte una de las líneas será

$$Q \leftarrow 1 - P$$

o sea

$$Q \leftarrow 2 \times D[2] \div OD[1]$$

de donde el valor de  $PI$  será aproximadamente

$$PI \leftarrow (2 \times N \times D[2]) \div K \times D[1]$$

donde  $K$  es el número de ocasiones que la aguja intersekte a alguna recta.

La función definida en *APL* que hace experimentalmente la simulación de arrojar la aguja y contar el número de éxitos es:

$$\forall Z \leftarrow N \text{ AGUJA } D; A; B$$

$$[1] \quad A \leftarrow D[1] \times (\overline{1} + ?Np1 + N) \div N$$

$$[2] \quad B \leftarrow D[2] \times 0.5 \times 2 \times \overline{0.5} + (\overline{1} + ?Np1 + N) \div N$$

$$[3] \quad Z \leftarrow (2 \times N) \div (\div / D) \times + / (B \geq A) \vee D[1] \leq A + B$$

∇

$N$  es un escalar y es el número de veces que se lanzará la aguja.

$D$  es un vector de dos dimensiones. Su primera coordenada ( $D[1]$ ) es la distan-

cia entre las rectas y su segunda coordenada( $D[2]$ ) es la longitud de la aguja.

$A$  y  $B$  son variables locales; ambos son vectores de dimensión  $N$ .

En el paso [1] al vector se le asignan  $N$  coordenadas cada una de ellas aleatoria, distribuída uniformemente en el intervalo  $0 \leq A \leq D[1]$ .

En el paso [2] al vector  $B$  se le asignan  $N$  coordenadas que son el resultado de calcular el coseno de  $N$  números aleatorios distribuídos uniformemente en el intervalo  $(0-0.5) \leq B \leq 0.5$  y multiplicar esto por  $D[2] \times 0.5$ .

En el paso [3] se calcula cuantas veces se intersecaron las rectas con la aguja. Basta que se cumplan cualesquiera de las siguientes dos condiciones:

$$D[1] \leq A + B$$

$$B \geq A$$

Se unen estas dos condiciones mediante la disyunción creando un vector lógico el cual se reduce con la suma, lo que nos da el número de intersecciones. Eso se multiplica por la reducción del vector  $D$  y se toma su inverso multiplicativo, lo que es equivalente a tener  $D[2] \times K \times D[1]$ .

Ahora sólomente se multiplica por  $2 \times N$  para obtener el valor aproximado de  $\pi$ .

Hé aquí algunos ensayos:

100 AGUJA 1 1  
3.50877193

1000 AGUJA 2 1  
3.134796238

1000 AGUJA 2 1  
3.448275862

1000 AGUJA 3 2  
3.384094755

1000 AGUJA 3 2  
3.0792991763

2000 AGUJA 3 2  
3.208985158  
3000 AGUJA 3 2  
3.098373354  
3000 AGUJA 3 2  
3.270645953  
3000 AGUJA 3 2  
3.076923077  
3000 AGUJA 3 2  
3.115264798  
3000 AGUJA 2 1  
3.128258603  
3000 AGUJA 2 1  
3.10880829  
3000 AGUJA 2 1  
3.092783505  
3000 AGUJA 2 1  
3.089598352  
3000 AGUJA 2 1  
3.015075377  
3000 AGUJA 2 1  
3.232758621  
3000 AGUJA 2 1  
3.099173554  
3000 AGUJA 2 1  
3.024193548  
3000 AGUJA 2 1  
3.080082136

3000 AGUJA 2 1

3.171247357

Como se notará, los valores obtenidos todos oscilan alrededor del valor 3.1416.

Los problemas tratados con los métodos Montecarlo son de dos tipos: Los llamados probabilísticos y los llamados determinísticos, dependiendo si están relacionados directamente o no con el comportamiento de procesos aleatorios.

En el caso probabilístico se trata principalmente de analizar números aleatorios, escogidos de tal manera que simulen los procesos aleatorios reales del problema original y nos infieran la solución aproximada.

A partir de la segunda guerra mundial los métodos Montecarlo han tenido gran desarrollo y se han aplicado mucho a problemas físicos tales como comportamiento de partículas, difusión de rayos, etc. El objetivo de esta tesis no es tratar este tipo de problemas, aunque el lector debe de estar enterado de estas aplicaciones.

La manera como se determinan los resultados en general es a través de la observación del proceso aleatorio y el cálculo de sus características estadísticas, que son bastante cercanos a los parámetros requeridos.

Por ejemplo, el resultado buscado  $X$  puede ser la esperanza matemática de cierta variable aleatoria  $V$ .  $X$  se podría calcular aproximadamente generando  $N$  valores de  $V$  en forma independiente (entonces  $V$  sería un vector de dimensión  $N$ ). Luego se calcularía el promedio

$$\bar{V} = \sum V / N$$

De acuerdo a la ley de los grandes números  $\bar{V}$  es aproximadamente igual a la esperanza de  $V$  la cual es  $X$ . Mientras más grande sea  $N$ , mejor aproximación se obtendrá.

Los métodos Montecarlo tienen sus ventajas y sus desventajas.

Quizás una de las más grandes ventajas de estos métodos es su facilidad de manejo aún en espacios multidimensionales, donde otro tipo de métodos numéricos se vuelven muy complicados.

Como en cualquier problema referente a métodos numéricos, también el error es de gran importancia en los métodos Montecarlo. La manera más fácil, desde el punto de vista probabilístico, para reducir el error, es aumentar el número de eventos. Esto tiene una gran desventaja y es que el número debe aumentarse en una proporción geométrica para obtener una exactitud mayor.

Otra manera más efectiva de hacerlo es cambiar el problema original por otro equivalente de tal manera que el error es reducido sensiblemente. Hay muchos métodos para hacer esto y son conocidos como métodos de reducción de varianza.

Los métodos Montecarlo empezaron a usarse en investigación en los trabajos relacionados con la boma atómica durante la segunda guerra mundial. La aplicación era una simulación directa de los problemas probabilísticos referentes a la difusión aleatoria de neutrones en materiales de fisión.

Con el incremento de la velocidad en las computadoras, los métodos de reducción de varianza tienden a ser menos interesantes, pero no por eso se elimina su utilidad.

En los últimos años los métodos Montecarlo se han hecho muy populares porque en muchos problemas son los más eficientes y en otros los únicos disponibles.

En los siguientes capítulos se tratará de dar una visión de algunas aplicaciones de métodos Montecarlo a problemas específicos.

## NUMEROS ALEATORIOS

Al plantearse un problema a resolver por medio de los métodos Montecarlo, llega un momento en que se tiene que hechar mano de un conjunto de números aleatorios, que necesariamente deben de cumplir con ciertos requisitos mínimos en cuanto a sus características estadísticas, como por ejemplo, su uniformidad en la distribución en cierto intervalo (generalmente el  $[0,1]$ ).

Muchas veces surge cierta confusión en cuanto a lo que significa número aleatorio. Estrictamente hablando, un número aleatorio es el producto de un proceso aleatorio impredecible para generarlo.

Este concepto estricto podría traer muchos problemas de orden práctico a la hora de generar números aleatorios, pues serían necesarios prácticamente un número infinito de ellos, debido a que para cada utilización de ellos se tendría que generar unos nuevos.

Generalmente los números aleatorios no son generados de esta manera. Existen una serie de métodos, que siendo predecibles, nos generan sucesiones de números llamados pseudo-aleatorios los cuales, una vez analizadas algunas de sus características estadísticas, resulta que se comportan como números aleatorios generados por procesos completamente aleatorios e impredecibles.

Es necesario destacar la pequeña trampa que se hace introduciendo un método determinístico para generar números que se comporten como si fueran aleatorios. Se sacrifica el idealismo para poder ganar efectividad.

La mayoría de las computadoras, al ser requeridas con números aleatorios siguen una regla bien específica, generando una sucesión de números pseudo-aleatorios.

A continuación se expondrán algunos métodos propuestos para la generación de sucesiones de números pseudo-aleatorios.

Antes de entrar en materia permítaseme definir dos nuevas funciones primiti-



vas diádicas de APL. Estas son la función 'ι'(decodificación) y 'τ'(codificación). Veamos primero la función 'ι'.

$AιB$

$A$  y  $B$  son dos vectores de la misma dimensión, o bien  $A$  puede ser un escalar y  $B$  un vector (también hay extensiones a otras estructuras que de momento no nos interesan). El resultado numérico de esto es la conversión a sistema decimal de  $B$  expresado en base numérica posicional de los elementos de  $A$ . La base puede ser constante (un escalar) o una mezcla de valores. Algunos ejemplos ilustrarán más claramente su uso:

10 10 10ι5 4 2

542

10ι3 2 6 7

3267

2ι1 1 1 1 0 1

61

Veamos la función 'τ'

$AτB$

$A$  debe ser un vector y  $B$  un escalar. el resultado es la representación de  $B$  en el sistema numérico que tiene como base los elementos del vector  $A$ , todo esto en un vector. Ejemplos:

10 10τ28

2 8

10 10 10τ1357

3 5 7

(5ρ2)τ18

1 0 0 1 0

Una vez definidas estas funciones, regresemos a nuestros métodos para generar números pseudo-aleatorios. El primer método fue propuesto por Von Neumann y Metro

polis y es el llamado método de los 'medios cuadrados'.

El método consiste en tomar un número cualquiera de cuatro dígitos, por ejemplo 4817.

$$A \leftarrow 4817^2$$

A

23203489

$$A \leftarrow 10.01 \times A$$

A

232034

$$A \leftarrow (4 \rho 10) \text{TA}$$

A

2 0 3 4

$$A \leftarrow 10.1A$$

A

2034

Nótese que lo que se ha hecho únicamente es elevar el número original al cuadrado y al resultado se le han suprimido los primeros dos dígitos y los últimos dos dígitos obteniendo un nuevo número pseudo-aleatorio. A este número se le hace el mismo proceso

$$A \leftarrow A^2$$

A

4137156

$$A \leftarrow 10.01 \times A$$

A

41371

$$A \leftarrow (4 \rho 10) \text{TA}$$

A

1 3 7 1

$A \leftarrow 10.1A$

A

1371

Abreviando este proceso, la expresión de APL para obtener un nuevo número pseudo-aleatorio sería:

$A \leftarrow 10.1(4\rho 10)\tau 10.01 \times A * 2$

Creemos una función para hacer este proceso a la cual llamaremos ALEATORIO1:

$\nabla N$ -ALEATORIO1 M

[1]  $\rightarrow 2 \times N \neq 0$

[2]  $\boxed{N} \leftarrow 10.1(4\rho 10)\tau 10.01 \times M * 2$

[3]  $(N-1)$  ALEATORIO1 M

$\nabla$

$N$  es el total de números pseudo-aleatorios que queremos nos genere la función y  $M$  es un entero de 4 dígitos que sirve como partida.

En el paso [1] se pregunta si  $N$  es distinto de cero. En caso afirmativo se pasa al renglón [2], de lo contrario se va al renglón [0], que es equivalente a salirse de la función.

El paso [2] empieza con el símbolo ' $\boxed{N}$ ' que significa que se va a sacar al exterior un valor numérico, en este caso, el nuevo valor de  $M$ . Aquí se está calculando el nuevo número pseudo-aleatorio y en el paso [3] la función recurre a sí misma con argumento izquierdo  $N-1$  (ahora hay que generar  $N-1$  números aleatorios) y argumento derecho  $M$  (que acaba de ser modificada).

La recursividad es un concepto importantísimo de APL, que resulta más elegante que las iteraciones tradicionales de otros lenguajes. Como un breve ejemplo, definiremos la función que nos calcule el factorial de un número natural recursivamente.

VZ←FACTORIAL N

[1] →2+2×N=0

[2] Z←N×FACTORIAL N-1

[3] →0

[4] Z←1

V

Sabemos que

$(!N)← →N×!N-1$

por lo tanto es fácil definir el factorial

FACTORIAL 1

1

FACTORIAL 0

1

FACTORIAL 6

720

Ahora hagamos algunos ejemplos de la función ALEATORIO1.

50 ALEATORIO1 4817

2034

1371

8796

3696

6604

6128

5523

5035

3512

3341

1622

6308

7908

5364

7724

6601

5732

8558

2393

7264

7656

6143

7364

2284

2166

6915

8172

7815

742

5505

3050

3025

1506

2680

1824

3269

6863

1007

140

196

384

1474

1726

9790

8841

2504

2700

2900

4100

8100

50 ALEATORIO1 1011

221

488

2381

6691

7694

1976

9045

8120

9344

3103

6286

5137

3887

1087

1815

2942

6553

9418

6987

8181

9287

2483

1652

7291

1586

1586

5153

5534

6251

750

5625

6406

368

1354

8333

4388

2545

4770

7529

6858

321

1030

609

3708

7492

1300

6900

6100

2100

4100

8100

50 ALEATORIO1 5739

9361

6283

4760

6576

2437

9389

1533

3500

2500

2500

2500

2500

2500

2500

2500



BIBLIOTECA  
INSTITUTO DE ECOLOGIA  
UNAM



2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

2500

Nótese como en el último caso se repite indefinidamente el número 2500, lo cual descarta a este método como algo efectivo para el requerimiento de uniformidad en cuanto a la distribución.

Otra manera de obtener la sucesión anterior es como sigue:

$$A \leftarrow 4817 * 2$$

A

23203489

$$A \leftarrow 10.01 * A$$

A

232034

$$A \leftarrow (2p10) * A$$

A

3 4

$$A \leftarrow 10.1A$$

A

34

$$B \leftarrow 4817 * 2$$

B

23203489

$B \leftarrow 10.0001 \times B$

$B$

2320

$B \leftarrow (2p10) \uparrow B$

$B$

2 0

$B \leftarrow 10 \downarrow B$

$B$

2000

$A \leftarrow A + B$

$A$

2034

La función en APL quedaría así:

$\nabla N \text{ ALEATORIO2 } M$

[1]  $\rightarrow 2 \times N \neq 0$

[2]  $\leftarrow M + (10 \downarrow (2p10) \uparrow 10.01 \times M * 2) + 100 \times 10 \downarrow (2p10) \uparrow 0.0001 \times M * 2$

[3]  $(N-1) \text{ ALEATORIO2 } M$

$\nabla$

Llamemos a la función.

50 ALEATORIO2 4817

2034

1371

8796

3696

6604

6128

5523

5035

3512

3341

1622

6308

7908

5364

7724

6601

5732

8558

2393

7264

7656

8143

7364

2284

2166

6915

8172

7815

742

5505

3050

3025

1506

2680

1824

3269

6863

1007

140

196

384

1474

1726

9790

8441

2504

2700

2900

4100

8100

Otro método que se propone y que es muy usado es el llamado método de congruencias; consiste en que dado  $M$ , un vector de tres coordenadas se utiliza la siguiente relación de recursión:

$$M[2] \leftarrow M[3] + M[1] \times M[2]$$

La función queda definida así:

$VN$  ALEATORIO3  $M$

[1]  $\rightarrow 2 \times N \neq 0$

[2]  $[M[2] \leftarrow M[3] + M[1] \times M[2]$

[3]  $(N-1)$  ALEATORIO3  $M$

$\nabla$

Hagamos algunas pruebas.

50 ALEATORIO3 16807, 16807,  $\lceil 1+2*31$

282475249

1622650073

984943658

1144108930

470211272

101027544

1457850878

1458777923

2007237709

823564440

1115438165

1784484492

74243042

114807987

1137522503

1441282327

16531729

823378840

143542612

896544303

1474833169

1264817709

1998097157

1817129560

1131570933

197493099

1404280278

893351816

1505795335

195489097

1636807926

563613512

101929267

1580723810

704877633

1358580979

1624379149

2128236579

784558821

530511967

2110010672

1551901393

1617819336

1399125485

156091745

1366425228

1899894091

585640194

937186357

1646035001

50 ALEATORIO3 125,16807,2\*42

2100875

262609375

32826171875

4103271484375

2735540258811

3292950996367

2600549013203

4011231339783

26615207019

3626900877375

2446237628099

2314494246199

3438757553115

3234182562287

4050587775411

548123149415

2544696010315

1427652489887

2534700791715

178250164887

291038055355

1195384830543

4287568951443

3782491086791

2220409160747

474214893823

2102257083523

3297391285239

3155585122203

3022000787119

3916144946035

1334955521831



4141719318027

3143472954207

1507979787619

3779519986007

1849021562747

2429276765967

194386479699

2308077406855

2636652635115

4126137567679

1195754160707

4333735221943

757181877083

2288757902191

221714552115

1326039947751

3027272558027

177069798431

50 ALEATORIO3 23,16807,10\*8

386561

8890903

4490769

3287687

75616801

39186423

1287729

29617767

81208641

67798743

59371089

65535047

7306081

68039863

64916849

93087527

41013121

43301783

95941009

6643207

52793761

14256503

27899569

41690087

58872001

54056023

43288529

95636167

99631841

91532343

5243889

20609447

74017281

2397463

55141649

68257927

69932321

8443383

94197809

66549607

30640951

4742103

9068369

8572487

97167201

34845623

1449329

33334567

66695041

33985943

Hay autores que prefieren la ecuación recursiva

$$M[3] \leftarrow M[4] | M[1] + M[2] \times M[3]$$

lo cual nos da la función

$$\forall N \text{ ALEATORIO}_4 M$$

[1]  $\rightarrow 2 \times N \neq 0$

[2]  $[M[3] \leftarrow M[4] | M[1] + M[2] \times M[3]$

[3]  $(N-1) \text{ ALEATORIO}_4 M$

$\nabla$

$$50 \text{ ALEATORIO}_4 56349, 16807, 16807, \bar{1} + 2 * 31$$

282531598

422280418

1981071987

1290478770

1639392686

1057738941

548807870

371663574

1667299091

1929252730

95103406

675167623

238705362

421622887

1667366705

918157581

1804516521

1745161862

603820257

1546883673

1036917878

649036490

1286900666

1631740874

1302753477

1781963123

647323548

414771883

339175768

1105589987

1622454014

1984803688

1732151714

985594815

1350742743

883705513

441710688

2128105533

729608695

401768844

840431289

1128784253

606458922

797769741

1375685115

1330840552

1395030308

47984959

1176894637

1743831538

El usar multiplicación en el método congruencial es un poco tardado en cuanto a tiempo de proceso. Un método congruencial con sumas en vez de multiplicaciones es más rápido.

Se puede hacer esto mediante la sucesión de Fibonacci, que queda mejor ilustrada mediante la siguiente función:

$\forall N \text{ ALEATORIO } M; Z$

[1]  $\rightarrow 2 \times N \neq 0$

[2]  $\square Z \leftarrow M[3] \mid M[1] + M[2]$

[3]  $M[1] \leftarrow M[2]$

[4]  $M[2] \leftarrow Z$

[5]  $(N-1)$  ALEATORIOS  $M$

▽

50 ALEATORIOS 0,1,2\*44

1

2

3

5

8

13

21

34

55

89

144

233

377

610

987

1597

2584

4181

6765

10946

17711

28657

46368

75025

121393

196418

317811

514229

832040

1346269

2178309

3524578

5702887

9227465

14930352

24157817

39088169

63245986

102334155

165580141

267914296

433494437

701408

1134903170

1836311903

2971215073

4807526976

7778742049

12586269025

20365011074

Nótese como los números son todo el tiempo crecientes y eso es debido a que se escogió

$M[1] \leftarrow 0$

$M[2] \leftarrow 1$

Hagámoslo con otros valores.

50 ALEATORIOS (2\*20),(2\*30),2\*44

1074790400

2148532224

3223322624

5371854848

8595177472

13967032320

22562209792

36529242112

59091451904

95620694016

154712145920

250332839936

405044985856

655377825792

1060422811648

1715800637440

2776223449088

4492024086528

7268247535616

11760271622144



1436333113344  
13196604735488  
14632937848832  
10237356539904  
7278108344320  
17515464884224  
7201387184128  
7124666023936  
14326053208064  
3858533187584  
592400351232  
4450933538816  
5043333890048  
9494267428864  
14537601318912  
6439682703360  
3385097977856  
9824780681216  
13209878659072  
5442473295872  
1060165910528  
6502639206400  
7562805116928  
14065444323328  
4036063395840  
509321674752  
4545385070592

5054706745344

9600091815936

14654798561280

Aquí ya aparece mejor la situación.

El sistema APL internamente el método de congruencias con multiplicaciones para la función primitiva '?'. En este caso

$M[1] \leftarrow 16807$

$M[2] \leftarrow 16807$

$M[3] \leftarrow \bar{1} + 2 * 31$

Si deseamos simular la función selección al azar lo podemos hacer mediante la siguiente función:

$\forall N$  ALEATORIO6 M;A

[1]  $\rightarrow 2 * N \neq 0$

[2]  $A \leftarrow M[3] | M[1] * M[2]$

[3]  $1 + M[4] | M[2] \leftarrow A$

[4]  $(N-1)$  ALEATORIO6 M

$\nabla$

$M[4]$  actúa como el argumento de la función '?'.  
50 ALEATORIO6 16807,16807, ( $\bar{1} + 2 * 31$ ), 50

50

24

9

31

23

45

29

24

10

41

16

43

43

38

4

28

30

41

13

4

20

10

8

11

34

50

29

17

36

48

27

13

18

11

34

30

50  
30  
22  
18  
23  
44  
37  
36  
46  
29  
42  
45  
8  
2

50 ALEATORIO6 16807,16807,(-1+2\*31),100

50  
74  
59  
31  
73  
45  
79  
24  
10  
41  
66  
93

43

88

4

28

30

41

13

4

70

10

58

61

34

100

79

27

13

68

11

34

80

50

80

22

68

73

94

37

86

46

29

92

95

58

2

La principal exigencia en cuanto a estos métodos es que nos proporcionen números pseudo-aleatorios que se acerquen lo más posible a una distribución uniforme en cierto intervalo. La prueba que más comúnmente se hace con estos números es la llamada ' $\chi$  cuadrada' para evaluar que tanto porcentaje de error se puede esperar si suponemos que estos números están distribuidos uniformemente. Las funciones *ALEATORIO1* y *ALEATORIO2* no las consideramos porque ya vimos que son repetitivas en cierto momento.

Haremos la prueba con los últimos cuatro métodos y con la función monádica '?' de la máquina.

Primero definiremos la función que nos entrega ' $\chi$  cuadrada'.

$Z \leftarrow JIC; E$

[1]  $Z \leftarrow ((A-E)*2) \div E \leftarrow (+/A) \div \rho A$

$\nabla$

$A$  es un vector tal que  $\rho A$  es el número de sub-intervalos iguales en que está dividido el intervalo original, y en cada coordenada tiene como valor la cantidad de números aleatorios que cayeron en ese sub-intervalo.  $E$  es el valor esperado, o sea  $(+/A) \div \rho A$ . El valor proporcionado por esta función es ' $\chi$  cuadrada' con  $1 + \rho A$  grados de libertad. El valor de confiabilidad se busca en las tablas de ' $\chi$  cuadrada' con  $1 + \rho A$  grados de libertad.

Para generar el vector  $A$  se definieron distintas funciones dependiendo del método. Estas se tuvieron que hacer iterativas debido a escasez de memoria en la

computadora en la que fueron probadas (IBM 370-155). Hélas aquí:

∇N PR3 M;B

[1]  $A \leftarrow M[4] \rho 0$

[2]  $A[B] \leftarrow A[B + (M[3] \div M[4]) \div 1 + M[2] \leftarrow M[3] | M[1] \times M[2]] + 1$

[3]  $\rightarrow 2 + 2 \times 0 = N \leftarrow N - 1$

[4] A

∇

∇N PR4 M;B

[1]  $A \leftarrow M[5] \rho 0$

[2]  $A[B] \leftarrow A[B + (M[4] \div M[5]) \div 1 + M[3] \leftarrow M[4] | M[1] + M[2] \times M[3]] + 1$

[3]  $\rightarrow 2 + 2 \times 0 = N \leftarrow N - 1$

[4] A

∇

∇N PR5 M;Z;B

[1]  $A \leftarrow M[4] \rho 0$

[2]  $A[B] \leftarrow A[B + (M[3] \div M[4]) \div 1 + Z \leftarrow M[3] | M[1] + M[2]] + 1$

[3]  $M[1] \leftarrow M[2]$

[4]  $M[2] \leftarrow Z$

[5]  $\rightarrow 2 + 4 \times 0 = N \leftarrow N - 1$

[6] A

∇

∇N PR6 M;B

[1]  $A \leftarrow M[5] \rho 0$

[2]  $A[B] \leftarrow A[B + (M[4] \div M[5]) \div 1 + M[4] | M[2] \leftarrow M[3] | M[1] \times M[2]] + 1$

[3]  $\rightarrow 2 + 2 \times 0 = N \leftarrow N - 1$

[4] A

∇

∇N PR7 M;B

- [1]  $A \leftarrow M[2]p0$
- [2]  $B \leftarrow (M[1] \neq M[2]) \neq ?Np.M[1]$
- [3]  $A[M[2]] \leftarrow +M[2] = B$
- [4]  $\rightarrow 3 + 2 \times 0 = M[2] \leftarrow M[2] - 1$
- [5]  $A$

v

En todas las funciones el último elemento del vector  $M$  es el número de sub-intervalos que se van a considerar. He aquí algunas pruebas ( $NC$  es el nivel de confiabilidad):

10000 PR3 16807,16807,(-1+2\*31),100

105	91	95	104	90	106	102	91	105	103	101	89	99	99	111	93	106	116	96	97
97	97	109	103	102	100	98	96	104	92	79	95	99	99	98	81	105	109	108	85
98	111	89	94	99	103	90	102	103	112	101	128	112	83	113	98	93	107	102	112
102	104	88	89	110	112	91	89	104	100	109	103	82	81	97	97	113	103	91	88
95	126	99	90	100	96	125	111	93	91	95	81	89	116	98	101	101	109	122	104

JIC

96.26

NC+0.56

10000 PR4 56349,16807,16807,(-1+2\*31),100

108	85	112	104	109	109	107	103	90	98	95	91	98	107	104	116	82	95	91	120
106	107	87	99	97	108	83	116	95	96	97	103	105	111	94	94	97	101	109	104
92	93	107	103	100	108	115	92	92	117	114	88	104	89	93	90	88	99	109	94
103	99	97	92	114	111	99	103	111	90	90	93	88	110	87	92	95	103	95	98
107	94	113	105	88	126	110	110	101	95	92	89	102	87	87	101	100	108	102	93

JIC

83.46

NC+0.87



10000 PR5 0,1,2,(2\*44),100

150 105 102 107 105 113 83 90 96 112 86 89 82 115 105 101 97 107 112 86  
100 118 93 105 102 100 101 95 98 96 95 106 106 107 102 82 121 109 122 85  
117 80 90 96 106 99 89 115 108 90 96 113 113 90 103 98 92 105 103 107  
91 93 111 95 98 94 94 104 100 97 101 92 93 96 95 91 112 90 100 91  
91 101 102 90 86 100 107 108 88 88 110 100 99 99 89 103 95 113 107 100

JIC

111.26

NC+0.19

10000 PR5 (2\*20),(2\*30),(2\*44),100

117 95 107 99 81 107 99 106 98 96 97 104 99 101 90 94 99 110 99 97  
111 110 113 106 98 89 104 121 97 84 95 100 99 102 101 104 95 104 113 93  
110 108 102 104 101 105 104 86 107 96 101 104 89 108 102 84 100 92 90 98  
107 96 106 97 103 107 87 79 100 92 87 100 97 89 104 91 112 94 103 99  
113 92 98 96 114 88 87 99 113 95 105 106 107 96 114 111 97 101 93 100

JIC

66.88

NC+0.99

10000 PR6 16807,16807,( $\bar{1}+2*31$ ),100,100

114 101 92 101 89 107 96 99 93 93 105 94 108 97 106 111 104 99 86 117  
121 88 95 109 98 83 93 103 97 94 81 99 98 105 109 93 100 96 91 106  
99 75 88 98 108 96 93 102 113 93 98 101 103 101 101 87 106 120 108 92  
103 113 107 110 93 95 102 90 95 117 98 108 87 94 93 112 96 99 92 130  
90 113 76 107 102 108 89 114 93 100 92 108 106 100 109 107 117 96 88 98

JIC

92.96

NC+0.65

10000 PR7 ( $\bar{1}+2*31$ ),199

105 91 95 104 90 106 102 91 105 103 101 89 99 99 111 93 106 116 96 97  
 97 97 109 103 102 100 98 96 104 92 79 95 99 99 98 81 105 109 108 85  
 98 111 89 94 99 103 90 102 103 112 101 128 112 83 113 98 93 107 102 112  
 102 104 88 89 110 112 91 89 104 100 109 103 82 81 97 97 113 103 91 88  
 95 126 99 90 100 96 125 111 93 91 95 81 89 116 98 101 101 109 122 104

JIC

96.26

NC=0.56

10000 PR7 100 100

102 108 108 80 89 93 109 116 101 86 98 112 108 99 109 95 117 91 105 110  
 109 98 107 84 101 94 103 99 98 89 95 110 112 111 94 98 110 100 95 85  
 102 108 102 91 87 106 98 101 89 109 98 105 97 103 88 98 98 110 112 98  
 101 106 121 87 92 109 95 98 85 108 102 86 84 102 114 87 101 112 114 108  
 93 93 99 111 87 101 95 96 100 105 99 118 88 112 89 93 94 80 107 100

JIC

84.68

NC=0.85

Obsérvese en PR5 como al variar los números iniciales aumenta de manera notable la confiabilidad.

Por último, hagamos una tabla de los valores obtenidos.

Método	función	JIC	NC
cong. 'x'	10000 PR3 16807,16807, ( $\bar{1}+2*31$ ),100	96.26	0.56
cong. '+' y 'x'	10000 PR4 56349,16807,16807, ( $\bar{1}+2*31$ ),100	83.46	0.87
cong. '+'	10000 PR5 0,1,(2*44),100	111.26	0.19
cong. '+'	10000 PR5 (2*20),(2*30),(2*44),100	66.88	0.99
simulación APL	10000 PR6 16807,16807, ( $\bar{1}+2*31$ ),100,100	92.96	0.65
APL	10000 PR7 ( $\bar{1}+2*31$ ),100	96.26	0.56
APL	10000 PR7 100 100	84.68	0.85

El método de congruencias de sumas resultó tener una eficiencia sorprendentemente alta escogiendo los valores adecuados. El método intrínseco de la máquina se puede calificar de apenas regular con tendencia a no cumplir la hipótesis.

## INTEGRACION MULTIPLE.

El capítulo de integración es de interés primordial para todo aquel que tiene alguna relación con problemas referentes a métodos numéricos.

En el caso de integrales definidas en el plano conocemos una infinidad de métodos, como por ejemplo el de Simpson, el del trapecioide, etc. Estos pueden resultar muy efectivos y relativamente sencillos cuando se trata de el plano y algunas veces en el espacio de tres dimensiones. Sin embargo, cuando se pasa a dimensiones mayores, éstos se complican de tal manera que resulta punto menos que imposible manejarlos. entonces es cuando entra en acción con toda su valía el método Montecarlo que comparativamente no sufre mayores complicaciones por el incremento de la dimensión del espacio en el cual se va a integrar. Otro aspecto importante es que el tiempo de máquina y la memoria requerida aumentan en proporción aritmética, a diferencia de otros métodos que aumentan en proporción geométrica. Esto hace posible que cualquier computadora de tamaño mediano esté en posibilidad de manejarlo más o menos eficientemente.

Entrando más en materia, podemos distinguir principalmente dos métodos Montecarlo de integración, que son llamados comúnmente el método geométrico y el método crudo.

Para claridad en el problema, empezaremos a tratarlo para integrales de funciones de  $R$  en  $R$  para luego extenderlo a funciones de  $R^*N$  en  $R$ . Empezemos por el método geométrico:

Supongamos que existe cierta función  $F$  que va de un intervalo en los reales con codominio en  $R$  continua, y deseamos calcular su integral en el intervalo

$$V[1] \leq x \leq V[2]$$

Sabemos también que  $0 \leq F(x)$  en todo el intervalo de integración. Sea  $C$  una cota superior de  $F(x)$  en el intervalo de integración, con lo cual formamos el rectángulo que tiene como vértices los puntos

$$V[1], 0$$

$$V[2], 0$$

$$V[2], C$$

$$V[1], C$$

del cual sabemos que su área es

$$C \times V[2] - V[1]$$

Una vez considerado esto, generemos  $N$  parejas de números aleatorios  $S$  tal que

$$\rho S \rightarrow N, 2$$

y con las condiciones

$$V[1] \leq S[;1] \leq V[2]$$

$$0 \leq S[;2] \leq C$$

Ahora verifiquemos cuantos elementos de  $S[;2]$  cumplen la condición

$$S[;2] \leq F S[;1]$$

o sea, en total son

$$NP \leftarrow / S[;2] \leq F S[;1]$$

puntos que cumplen la condición, con lo cual el área aproximada bajo la curva quedará expresada de la siguiente manera:

$$(NP + N) \times C \times V[2] - V[1]$$

Pasemos ahora al caso del método crudo. Otra vez suponemos que  $F$  es una función de un intervalo contenido en  $R$  y codominio en  $R$ , continua y se desea calcular la integral en el intervalo  $V[1] \leq X \leq V[2]$ . La condición de que  $F$  sea mayor o igual que cero aquí no es necesaria; tampoco es necesario saber la cota superior de la función.

Generemos  $N$  números aleatorios  $S$  tales que

$$\rho S \rightarrow N, 1$$

$$V[1] \leq S \leq V[2]$$

Ahora tomemos

$$T \leftarrow F S$$



BIBLIOTECA  
INSTITUTO DE ECOLOGIA  
UNAM

(se calcula la función en cada elemento del vector columna  $S$ ), dándonos como resultado un vector renglón  $T$  tal que

$$\rho T \leftarrow \rightarrow N$$

Un estimador de la integral sería en este caso

$$(\rightarrow N) \times \rightarrow T \times V[2] - V[1]$$

o sea, la media aritmética de el producto de la función evaluada en los puntos aleatorios por la longitud del intervalo.

Antes de pasar al caso general en el espacio de  $N$  dimensiones, definiremos algunos conceptos nuevos de *APL*, que nos servirán para definir las funciones de integración.

El producto externo en *APL* tiene la siguiente sintaxis

$$C \leftarrow A . . f B$$

donde  $A$  y  $B$  son escalares o vectores y  $f$  es cualquier función primitiva escalar diádica. El resultado  $C$  es una matriz de dimensiones  $(\rho A), \rho B$ , tal que

$$C[I, J] \leftarrow A[I] f B[J]$$

De hecho, se generan tablas de las distintas funciones primitivas escalares diádicas.

Ejemplos:

$$A \leftarrow 16$$

$$B \leftarrow 2 + 14$$

$$A$$

1 2 3 4 5 6

$$B$$

3 4 5 6

$$A . . B$$

1 1 1 1

2 2 2 2

3 3 3 3

3 4 4 4

3 4 5 5

3 4 5 6

$B \cdot \Gamma A$

3 3 3 4 5 6

4 4 4 4 5 6

5 5 5 5 5 6

6 6 6 6 6 6

$A \cdot \leq B$

1 1 1 1

1 1 1 1

1 1 1 1

0 1 1 1

0 0 1 1

0 0 0 1

Ahora veamos el concepto de producto interno, cuya sintaxis general es:

$A \cdot f \cdot g \cdot B$

donde  $f$  y  $g$  deben de ser funciones escalares primitivas diádicas. El producto interno funciona para las estructuras numéricas que cumplan alguna de las siguientes condiciones:

- 1.-  $A$  escalar y  $B$  cualquier estructura.
- 2.-  $A$  cualquier estructura y  $B$  escalar.
- 3.-  $A$  cualquier estructura y  $B$  cualquier estructura tal que la última dimensión de  $A$  coincida con la primera dimensión de  $B$ .

El resultado de esta función es en el caso de escalares o dos vectores

$f/A \cdot g \cdot B$

En el caso en que  $A$  sea una matriz y  $B$  un vector o un escalar, es el vector  $C$  tal que

$$C[I] \leftarrow f/A[I;] \text{ g } B$$

Si  $A$  es un vector o un escalar y  $B$  una matriz, será el vector  $C$ , tal que

$$C[J] \leftarrow f/A \text{ g } B[:,J]$$

En el caso en que ambas sean matrices, dará como resultado la matriz  $C$  tal que

$$C[I;J] \leftarrow f/A[I;] \text{ g } B[:,J]$$

En otras palabras, el resultado se obtiene tomando los vectores de la última dimensión de  $A$  y de la primera de  $B$ , aplicando entre ellos la función  $g$  y luego reduciéndolos mediante la función  $f$ .

Ejemplos

$$A \leftarrow 2 \text{ 3p } 16$$

$$B \leftarrow 3 \text{ 2p } \bar{1} + 16$$

$A$

1 2 3

4 5 6

$B$

0 1

2 3

4 5

$$A+.*B$$

16 22

34 49

$$A\lfloor.B$$

1 1

4 4



$$(A \vee > B) \wedge \vee A \wedge \< B$$

0 0

1 1

Pasemos ahora a ver las funciones de rotación que son ' $\phi$ ' y ' $\theta$ '; veamos los distintos casos.

$\phi B$  invierte el orden a través de la última dimensión de  $B$ .

$\theta B$  invierte el orden a través de la primera dimensión de  $B$ .

$\phi[C]B$  invierte el orden a través de la  $C$ -ésima (de adelante para atrás) dimensión de  $B$ .

$\theta[C]B$  invierte el orden de la  $C$ -ésima (contando de atrás para adelante) dimensión de  $B$ .

$A\phi B$  rota de acuerdo con  $A$ , a través de la última coordenada de  $B$ .

$A\theta B$  rota de acuerdo con  $A$ , a través de la primera dimensión de  $B$ .

$A\phi[C]B$  rota de acuerdo con  $A$ , a través de la  $C$ -ésima (de adelante para atrás) dimensión de  $B$ .

$A\theta[C]B$  rota de acuerdo con  $A$ , a través de la  $C$ -ésima (de atrás para adelante) dimensión de  $B$ .

$C$  debe ser un entero tal que pertenezca al vector

$$1 \rho \rho B$$

$A$  debe de ser un escalar entero o un vector entero, en cuyo caso debe coincidir  $\rho A$  con la dimensión de los planos a través de los cuales la rotación de es realizada.

Si  $A$  es positivo, la rotación se hace  $A$  lugares de izquierda a derecha, y en caso contrario serán  $-A$  lugares de izquierda a derecha. En espacios mayores, positivo sería de abajo para arriba y negativo de arriba para abajo.

En caso de que  $A$  sea un vector, la rotación se hace a través de la coordenada dada. Ejemplos:

$A \leftarrow 16$

$B \leftarrow 3 \text{ } 3 \text{ } 15$

$A$

1 2 3 4 5 6

$B$

1 2 3

4 5 1

2 3 4

$\phi A$

6 5 4 3 2 1

$\phi B$

3 2 1

1 5 4

4 3 2

$\ominus B$

2 3 4

4 5 1

1 2 3

$\phi \ominus B$

4 3 2

1 5 4

3 2 1

$\ominus \phi B$

4 3 2

1 5 4

3 2 1

4φA  
5 6 1 2 3 4  
-1φA  
6 1 2 3 4 5  
1θB  
4 5 1  
2 3 4  
1 2 3  
-1θB  
3 1 2  
1 4 5  
4 2 3  
-2: 1 2θB  
4 5 4  
2 3 3  
1 2 1

Ahora sí estamos listos para pasar al caso general de integración;comenzemos nuevamente con el método geométrico.Sea  $F$  una función continua que va de un intervalo en  $R^*I$  con codominio en  $R$ ,y que sea mayor o igual que cero en todos sus puntos.Sea  $V$  el intervalo en el cual se va a integrar.

$$I \leftarrow 0.5 \times \rho V$$

Las coordenadas impares de  $V$  tienen los puntos iniciales de los intervalos,y las pares los finales. $N$  es el número de puntos aleatorios que se van a considerar.

Generamos  $N$   $I$ -adas de vectores aleatorios que las ponemos en la matriz  $S$ .Las condiciones son:

$$\rho S \leftarrow \rightarrow N, I$$

$$\sqrt{1+2 \times 1 I} \leq S[;] \leq \sqrt{2 \times 1 I}$$

Calculamos la función  $F$  en cada renglón de la matriz  $S$ , lo cual nos da un vector de dimensión  $N$  llamado  $L$ .

$$L \leftarrow F S$$

$$\rho L \leftarrow N$$

Sabemos que

$$\lceil L$$

es una cota superior (la mínima) de los valores considerados en esta función.

Ahora generamos un vector  $T$  de  $N$  dimensiones tales que

$$0 \leq T \leq \lceil L$$

El área del hipercubo donde están encerradas las  $(I+1)$ -adas de números aleatorios es

$$(S/L) \times \sqrt{2 \times 1 I} - \sqrt{1+2 \times 1 I}$$

Ahora vemos cuantos puntos cumplen con la condición

$$T \leq L$$

mediante la expresión

$$+/T \leq L$$

con lo cual el área aproximada bajo la curva será

$$((+/T \leq L) \div N) \times (\lceil L) \times \sqrt{2 \times 1 I} - \sqrt{1+2 \times 1 I}$$

Definamos la siguiente función:

$$\forall Z \leftarrow V \text{ INTEGRAL } N; I; J; L; A; B$$

$$[1] \quad B \leftarrow (\sqrt{1+2 \times 1 I}) \ominus (I, I) \rho (\times \sqrt{2 \times 1 I} - \sqrt{1+2 \times 1 I}), (I \times (I + 0.5 \times \rho V) - 1) \rho 0$$

$$[2] \quad L \leftarrow F((N, I) \rho \sqrt{1+2 \times 1 I}) + ((\sqrt{1+2 \times 1 I}) \rho A + 1) \div A + 10 \times 9) + . \times B$$

$$[3] \quad Z \leftarrow ((\lceil L) \times \times / J) \times (+/L \geq (\lceil L) \times (\sqrt{1+2 \times 1 I}) \div A) \div N$$

∇

En el paso [1] se están definiendo:

$I$ , que es el espacio en el cual se hace la integración.

$J$ , que es el vector que contiene en cada coordenada la longitud de los  $I$  intervalos de integración.

$B$ , que es una matriz tal que

$$\rho B \leftarrow \rightarrow I, I$$

La diagonal principal de la matriz contiene los elementos del vector  $J$ , y las demás componentes son cero.

En el paso [2] se genera la matriz de dimensiones  $(N, I)$  de números aleatorios distribuidos en el intervalo  $[0, 1]$  mediante la fórmula

$$(\Gamma + ?(N, I) \rho A + 1) \div A \leftarrow 10 * 9$$

$A$  tiene valor de  $10 * 9$  por considerársele adecuado en cuanto a la variabilidad y no repetición de números aleatorios. Esto se multiplica internamente por la matriz  $B$  lo cual da como resultado una matriz de dimensiones  $(N, I)$  que sumada a la matriz

$$(N, I) \rho V \leftarrow 1 + 2 * I$$

da una matriz que tiene como característica el tener números aleatorios distribuidos uniformemente en los intervalos de integración.

Se calcula ahora la función a través de los renglones de esa matriz, y el valor obtenido (un vector de  $N$  coordenadas) se le asigna a  $L$ .

En el paso [3] se genera un vector de  $N$  coordenadas distribuidas al azar en el intervalo comprendido entre 0 y  $\Gamma/L$ . Se verifica cuantos puntos de los generados cumplen con la condición

$$L \geq (\Gamma/L) \times (\Gamma + ? N \rho A + 1) \div A$$

Se saca la proporción de puntos que cumplen la condición (dividiendo la cantidad anterior entre  $N$ ) y el área aproximada bajo la curva será el resultado de multiplicar la proporción obtenida por el área del hipercubo que es

$$(\Gamma/L) \times \times / J$$

Nótese que la función  $F$  recibe como argumento una matriz de dimensiones

(N,I) y entrega como resultado la evaluación de la función en cada renglón de la matriz original, todo esto en un vector de dimensión N.

Pasando al método crudo, nuevamente tenemos una función continua F que va de un intervalo en R\*I en R.V es el mismo vector que en el caso anterior. La función va a evaluar en cada iteración N[1] puntos hasta que el error relativo sea menor que N[2] (N[2] es un parámetro que se le da a la función).

Generamos N[1] I-adas de números aleatorios en la matriz S, cumpliendo ésta las mismas condiciones que en la función INTEGRAL. Evaluamos la función F en S; el vector que nos da como resultado lo multiplicamos por el hipervolumen del intervalo

$$\times / V[2 \times 1 I] - V[1 + 2 \times 1 I]$$

y tomamos la media aritmética de los valores. A este valor lo llamamos VA. Hacemos el mismo proceso con N[1] puntos más, o sea que sacamos la media aritmética de 2\*N[1] puntos y a este valor lo llamamos VN. Deseamos hacer este proceso hasta que

$$N[2] > 1 - VN \div VA \text{ (error relativo).}$$

Si se cumple la condición entonces VN será aproximadamente el valor de la integral.

La función APL queda así:

```

VZ ← V MONTE N; A; B; I; C; D; L; E; G
[1] L ← × / B + V[2 × 1 I] - A + V[1 + 2 × 1 I + 0.5 × ρ V]
[2] A ← ((D + N[1]), I) ρ A
[3] C ← + / L × F A + ((1 + ? B + (N[1], I) ρ G + 1) ÷ 10 * 9) + . × E + (1 × 1 + 1 I) ⊖ (I, I) ρ B, (I × I - 1) ρ 0
[4] → 4 + N[2] > |1 - ((C + + / L × F A + ((1 + ? B) ÷ G) + . × E) ÷ D + D + N[1]) ÷ [ + C ÷ D
[5] Z ← C ÷ D

```

V.

N es un vector de dos coordenadas: N[1], el número de puntos que se toman en

cada iteración y  $N[2]$ , el error relativo permitido.

En el paso [1],  $A$  es el conjunto de valores iniciales de los intervalos,  $B$  es el conjunto de longitudes de los intervalos y  $L$  es el área del hipercubo.

En el paso [2],  $A$  se convierte ahora en una matriz de dimensión  $(N[1], I)$  que tiene en cada renglón al vector  $A$  anterior.  $D$  va a ser el contador de puntos que ya se calcularon y tiene como valor inicial  $N[1]$ .

En el paso [3] se calcula la función de dimensión  $(N[1], I)$  con valores distribuidos aleatoriamente en los intervalos (análogo a la función *INTEGRAL*). Estos se multiplican por la longitud del intervalo y luego se suman entre sí.

En el paso [4] se calcula el valor anterior de la iteración ( $C/D$ ) y se saca al exterior. Luego se calcula el nuevo valor. Se computa el error relativo entre ambos y si todavía es grande, se regresa nuevamente al paso [4]; de lo contrario pasa al paso [5] que nos entrega el valor aproximado de la integral.

Ahora hagamos algunos ejemplos de funciones para integrar. Definimos

$\forall Z \leftarrow F X$

[1]  $Z \leftarrow (+/X)++/X \times 1 \phi X$

$\nabla$

Esta función pudo haber sido definida de una manera más larga, quizá más explícita, pero menos elegante, de la siguiente manera:

$\forall Z \leftarrow F X$

[1]  $Z \leftarrow X[;1] + X[;2] + X[;3] + (X[;1] \times X[;2]) + (X[;2] \times X[;3]) + X[;1] \times X[;3]$

$\nabla$

0 1 0 1 0 1 *INTEGRAL* 1000

2.267905493

0 1 0 1 0 1 *INTEGRAL* 1000

2.083415779

0 1 0 1 0 1 *INTEGRAL* 1000

2.307586723

0 1 0 1 0 1 *INTEGRAL* 1000  
2.232095211  
0 1 0 1 0 1 *INTEGRAL* 1000  
2.286803334  
0 1 0 1 0 1 *MONTE* 1000 .001  
2.219360252  
2.220167359  
0 1 0 1 0 1 *MONTE* 1000 .001  
2.293733594  
2.259509416  
2.258309999  
0 1 0 1 0 1 *MONTE* 1000 .001  
2.256107614  
2.269010145  
2.269507935  
0 1 0 1 0 1 *MONTE* 1000 .001  
2.244527606  
2.241410776  
2.242869497  
0 1 0 1 0 1 *MONTE* 1000 .001  
2.241540911  
2.252308281  
2.260095947  
2.250043395  
2.246259525  
2.249249984  
2.248852435



Definimos otra función.

$VZ \leftarrow F X$

[1]  $Z \leftarrow (X[;1] * 3) * X[;2] \div X[;1]$

$\nabla$

1 4 1 3 INTEGRAL 1000

4.48778916

1 4 1 3 INTEGRAL 1000

4.489130546

1 4 1 3 INTEGRAL 1000

4.814398316

1 4 1 3 INTEGRAL 1000

4.646934229

1 4 1 3 INTEGRAL 1000

4.095389188

1 4 1 3 MONTE 1000 .001

4.698217556

4.680222121

4.535958861

4.643584301

4.648114333

1 4 1 3 MONTE 1000 .001

4.44402494

4.508472146

4.453563061

4.470793682

4.518753195

4.526296739

4.549797693

4.541700296

4.500725856

4.486629422

4.490671627

1 4 1 3 MONTE 1000 .001

3.839674306

4.150458803

4.202434799

4.458368715

4.501272956

4.470541957

4.444326945

4.585626542

4.57289525

4.535504164

4.555146971

4.593049935

4.567328043

4.59965021

4.620999274

4.615418254

4.596959624

4.614555262

4.607589004

4.596193213

4.599151552

1 4 1 3 MONTE 1000 .001

4.941514433

4.756632332

4.703308843

4.739439853

4.72617692

4.623244916

4.624106647

1 4 1 3 MONTE 1000 .001

4.340731663

4.793470912

4.569352118

4.515193319

4.683677563

4.68146925

Ahora F será

$\nabla Z + F X$

[1]  $Z \leftarrow (X[;1]*2) \times ((X[;1]*2) - X[;2]*2) * 0.5$

$\nabla$

3 5 1 3 INTEGRAL 1000

0.09223618539

3 5 1 3 INTEGRAL 1000

0.09811868767

3 5 1 3 INTEGRAL 1000

0.08698661349

3 5 1 3 INTEGRAL 1000

0.08880498753

3 5 1 3 *INTEGRAL* 1000

0.08596656836

3 5 1 3 *MONTE* 1000 .001

0.08970812473

0.0877123973

0.09399090118

0.09331215767

0.09329856348

3 5 1 3 *MONTE* 1000 .001

0.09672950164

0.09352229061

0.09360906011

3 5 1 3 *MONTE* 1000 .001

0.09477315493

0.09250114841

0.0920124624

0.09232299036

0.09230713039

3 5 1 3 *MONTE* 1000 .001

0.08966036316

0.09091603355

0.09265431157

0.09256590303

3 5 1 3 *MONTE* 1000 .001

0.09314482225

0.09373217481

0.09352239173

0.0925562497

0.09255244359

$\nabla Z \leftarrow F_X$

[1]  $Z \leftarrow (X[;1]*2) \times ((X[;1]*2) - X[;2]*2) * 0.5$

$\nabla$

1 2 2 3 *INTEGRAL* 1000

0.08503438246

1 2 2 3 *INTEGRAL* 1000

0.08218782926

1 2 2 3 *INTEGRAL* 1000

0.08973210228

1 2 2 3 *INTEGRAL* 1000

0.08508937231

1 2 2 3 *INTEGRAL* 1000

0.08524044705

1 2 2 3 *MONTE* 1000 .001

0.08565707496

0.08556472944

0.08505391842

0.08499404563

1 2 2 3 *MONTE* 1000 .001

0.08548417859

0.08439848241

0.08433485892

1 2 2 3 *MONTE* 1000 .001

0.0835875198

0.0841320127

0.08430004907

0.08438376668

1 2 2 3 MONTE 1000 .001

0.08507754501

0.08462958386

0.08501320339

0.08497238425

1 2 2 3 MONTE 1000 .001

0.08549022438

0.08500148294

0.0848826987

0.08438242468

0.08486374386

0.08486450058

$\nabla Z \leftarrow F X$

[1]  $Z \leftarrow (X[;3] \div X[;2]) \times 10 X[;1] + X[;3]$

$\nabla$

$W \leftarrow 1, 2, 1, (00.5), 1, 00.5$

$W$

1 2 1 1.570796327 1 1.570796327

$W$  INTEGRAL 1000

0.2955615646

$W$  INTEGRAL 1000

0.2947666943

$W$  INTEGRAL 1000

0.3005075164

$W$  INTEGRAL 1000

0.2968376245

W INTEGRAL 1000

0.2945701029

W MONTE 1000 .001

0.2947875685

0.2966540327

0.296547645

W MONTE 1000 .001

0.2920649085

0.2939432739

0.2941641365

W MONTE 1000 .001

0.2944273484

0.2930739971

0.2928683467

W MONTE 1000 .001

0.2932296487

0.2922288614

0.2924991194

W MONTE 1000 .001

0.2952222318

0.2944765692

0.2942291525

La siguiente función tiene valores negativos, con lo cual no se cumple una de las condiciones establecidas por el método geométrico. Sólomente se darán ejemplos con el método crudo.

$\nabla Z = F X$

[1]  $Z = (X[;1] \times X[;3]) \times 2 \times X[;1] \times X[;2]$

∇

$W = (00, 5), (01), (01), (02), 0, 2$

$W$

1.570796327 3.141592654 3.141592654 6.283185307 0 2

$W$  MONTE 1000 .001

-0.4733622774

-1.112157323

-1.287678006

-1.470287613

-1.579566221

-1.425356528

-1.377631335

-1.500640241

-1.447327536

-1.355497143

-1.37230694

-1.387516106

-1.34810711

-1.33072771

-1.310185341

-1.274751043

-1.253930723

-1.268772847

-1.2812981

-1.253606923

-1.208392838

1.19389169

-1.22212895



-1.211456248

-1.224733895

-1.230229861

-1.225665198

-1.189205052

-1.208077702

-1.184162131

-1.167838805

-1.133782759

-1.15552611

-1.148953411

-1.15897474

-1.138208396

-1.122381313

-1.12913075

-1.101534291

-1.099303859

-1.086709426

-1.076606246

-1.091159155

-1.111216456

-1.113087743

-1.112743177

W MONTE 1000 .001

-2.039952156

-0.9215549358

-1.28580497

-1.161249321  
-1.210494996  
-1.136930957  
-1.201725416  
-1.096435854  
-1.121018659  
-1.029246498  
-1.013393488  
-1.066289823  
-1.092888539  
-1.1776451  
-1.101070484  
-1.073192746  
-1.098777024  
-1.042638661  
-1.068588184  
-1.078050508  
-1.063626107  
-1.075672456  
-1.073710152  
-1.081081413  
-1.071826542  
-1.134649251  
-1.143654856  
-1.170162856  
-1.192192143  
-1.194028182

1.170378006

1.217613284

1.241418259

1.211773308

1.195505746

1.183011059

1.201578645

1.197516924

1.219213659

1.237478941

1.267167428

1.258515532

1.272008856

1.276819593

1.26642071

1.273074294

1.25943535

1.243523184

1.240166631

1.227129754

1.205977204

1.2015132

1.231153293

1.248803521

1.253391062

1.26498065

1.25561704

1.23830454

1.254084141

1.242732886

1.233986227

1.233042144

W MONTE 1000 .001

2.234418836

2.073589874

1.862723659

1.569610288

1.57438547

1.736654438

1.806631141

1.753929362

1.636786583

1.535869564

1.423546373

1.480391543

1.488480447

1.452886633

1.457773924

1.506535626

1.540506517

1.560643892

1.467760364

1.421788062

1.341834981

1.367733503

-1.301911904

-1.313049017

-1.25700181

-1.266345116

-1.242497764

-1.204945936

-1.185727441

-1.17203089

-1.18986943

-1.226867858

-1.2009033

-1.223410335

-1.215242541

-1.237891502

-1.215600721

-1.23795011

-1.187968889

-1.17667673

-1.177251432

*W MONTE* 1000 .001

-1.259358534

-1.48351038

-1.103573853

-1.171170029

-1.415613961

-1.338019519

-1.1797672

-1.087037616  
-1.120720842  
-1.008634372  
-0.9740786707  
-1.018321445  
-1.011961091  
-0.9915955525  
-0.9501989829  
-0.9811791757  
-1.011153024  
-0.9907424825  
-1.048606507  
-1.052104127  
-1.059269662  
-1.086642049  
-1.08353081  
-1.052009106  
-1.058649624  
-1.077264088  
-1.06234493  
-1.102013925  
-1.088383689  
-1.055017595  
-1.056785969  
-1.064813495  
-1.106870609  
-1.100290544  
-1.077191842

-1.087809528

-1.101507764

-1.093434956

-1.083689368

-1.10370283

-1.089900909

-1.091566121

-1.099242444

-1.120592857

-1.109961408

-1.11020441

*W MONTE* 1000 .001

-0.6881778048

-0.9070459759

-1.17002106

-1.190654091

-1.150612576

-1.268838099

-1.20637847

-1.226997938

-1.162187643

-1.154092536

-1.150761192

-1.199464053

-1.263692438

-1.281818604

-1.255753723

-1.292426861  
-1.327028548  
-1.275895219  
-1.24718165  
-1.234395191  
-1.219741912  
-1.183971997  
-1.16998459  
-1.20169407  
-1.215643362  
-1.195596667  
-1.208044872  
-1.199227545  
-1.189134736  
-1.19670935  
-1.193568711  
-1.204213436  
-1.191874236  
-1.177314359  
-1.176172257

Como se mencionó anteriormente, también existen métodos de reducción de varianza. La razón por la cual no fueron incluidos aquí es que no se refieren en general a procedimientos universales. Depende de la función el método a seguir. La intención de esta tesis es referirse a métodos que sean más generales en cuanto a su aplicación.



CONCLUSIONES.

Al llegar a este punto, las conclusiones las podemos dividir en dos tipos: Las conclusiones referentes a los métodos Montecarlo y las que conciernen al lenguaje *APL*.

Lo que podemos decir con respecto a las primeras es que no pretendimos en esta tesis hacer un análisis minucioso y estricto de todos los métodos que tienen algo que ver con la generación de números aleatorios. La intención es que el lector se de cuenta que muchas veces cuando uno pretende resolver un determinado problema referente a análisis numérico, existe la opción de recurrir a los métodos Montecarlo. En general, este tipo de conceptos no están suficientemente difundidos y son raros los libros no especializados en el tema que lo tratan (como por ejemplo, los de análisis numérico).

La aplicación de los métodos Montecarlo está abierta, y sobre todo existen posibilidades ilimitadas en procesos de simulación, como por ejemplo, movimiento de partículas, epidemiología, sistemas ecológicos, etc.

En general, muchos fenómenos naturales se comportan en forma aleatoria, y a través de modelos estocásticos se pueden simular auxiliándose en los métodos Montecarlo.

Pasando a las conclusiones referentes a *APL*, se hicieron algunas comparaciones con otros lenguajes de uso común en programación. Recordemos la función *INTEGRAL* definida en el capítulo anterior; héla aquí:

$$\forall Z \leftarrow V \text{ INTEGRAL } N; I; J; L; A; B$$

$$[1] \quad B \leftarrow (\bar{1} \times \bar{1} + 1 \times I) \ominus (I, I) \rho (J + \sqrt{2 \times 1 I}] - \sqrt{1 + 2 \times 1 I}], (I \times (I + 0.5 \times \rho V) - 1) \rho 0$$

$$[2] \quad L \leftarrow F((N, I) \rho \sqrt{1 + 2 \times 1 I}] + ((\bar{1} + ?(N, I) \rho A + 1) \div A + 10 \times 9) + . \times B$$

$$[3] \quad Z \leftarrow ((\int / L) \times \times / J) \times (+ / L \geq (\int / L) \times (\bar{1} + ? N \rho A + 1) \div A) \div N$$

▽

También habíamos definido la función *F*

$$VZ \leftarrow F X$$

$$[1] Z \leftarrow (+/X)++/X \times 1 \Phi X$$

7

Los programas equivalentes fueron hechos en los lenguajes ALGOL, BASIC, FORTRAN y PL/I. A continuación se listan:

#### ALGOL

BEGIN

FILE BFBS (KIND=REMOTE);

REAL ARRAY J[0:2], V[0:5], C[0:2], I[0:999];

INTEGER I, K, N, NP, ALEA;

REAL MAX, AREA;

REAL PROCEDURE F(V1, V2, V3);

VALUE V1, V2, V3;

REAL V1, V2, V3;

F:=V1+V2+V3+V1\*V2+V1\*V3+V2\*V3;

READ(BFBS, /, FOR I:=0 STEP 1 UNTIL 5 DO V[I], N, ALEA);

FOR I:=0, 1, 2 DO

J[I]:=V[2\*I-1]-V[2\*I];

FOR I:=0 STEP 1 UNTIL N-1 DO

BEGIN

FOR K:=0, 1, 2 DO

C[K]:=V[2\*K]+J[K]\*RANDOM(ALEA);

L[I]:=F(C[0], C[1], C[2]);

IF MAX LSS L[I] THEN MAX:=L[I]

END;

FOR I:=0 STEP 1 UNTIL N-1 DO

IF MAX\*RANDOM(ALEA) LEQ L[I] THEN NP:=NP+1;

AREA:=1;

```
FOR I:=0,1,2 DO
AREA:=AREA*J[I];
AREA:=AREA*MAX;
WRITE(BFBS, <F20.10>, NP/N*AREA)
END.
```

BASIC

```
100 DIM J(3),V(6),L(1000),C(3)
200 MAT INPUT V
300 INPUT N,A1
400 FOR I=1 TO 3
500 J(I)=V(2*I)-V(2*I-1)
600 NEXT I
700 DEF FNF(V1,V2,V3)=V1+V2+V3+V1*V2+V1*V3+V2*V3
800 FOR I=1 TO N
900 FOR K=1 TO 3
1000 C(K)=V(2*K-1)+J(K)*RND(A1)
1100 NEXT K
1200 L(I)=FNF(C(1),C(2),C(3))
1300 IF M1 LT L(I) THEN 1500
1400 GO TO 1600
1500 M1=L(I)
1600 NEXT I
1700 FOR I=1 TO N
1800 IF M1*RND(A1) LE L(I) THEN 2000
1900 GO TO 2100
2000 N1=N1+1
2100 NEXT I
2200 A1=1
```

```
2300 FOR I=1 TO 3
2400 A1=A1*J(I)
2500 NEXT I
2600 A1=A1*M1
2700 PRINT N1/N*A1
2800 END
```

FORTRAN

```
FILE 1=BFBS,UNIT=REMOTE
DIMENSION RJ(3),V(6),C(3),RL(1000)
READ(1,/) (V(I),I=1,6),N,ALEA
DO 10 I=1,3
10 RJ(I)=V(2*I)-V(2*I-1)
DO 20 I=1,N
DO 30 K=1,3
30 C(K)=V(2*K-1)+RJ(K)*RANDOM(ALEA)
RL(I)=F(C(1),C(2),C(3))
20 IF (RMAX.LT.RL(I)) RMAX=RL(I)
DO 40 I=1,N
40 IF (RMAX*RANDOM(ALEA).LE.RL(I)) NP=NP+1
AREA=1.
DO 50 I=1,3
50 AREA=AREA*RJ(I)
AREA=AREA*RMAX
WRITE(1,60) FLOAT(NP)/FLOAT(N)*AREA
60 FORMAT(F20.10)
STOP
END
```

```
FUNCTION F(V1,V2,V3)
F=V1+V2+V3+V1*V2+V1*V3+V2*V3
RETURN
END
```

PL/I

INTEGRAL:PROCEDURE;

```
DECLARE I,K,N,NP,ALEA FIXED(10,0),RMAX,AREA,RJ(3),V(6),C(3),RL(1000);
```

```
GET LIST ((V(I) DO I=1 TO 6),N,ALEA);
```

```
DO I=1 TO 3;
```

```
  RJ(I)=V(2*I)-V(2*I-1);
```

```
END;
```

```
DO I=1 TO N;
```

```
  DO K=1 TO 3;
```

```
    C(K)=V(2*K-1)+RJ(K)*RANDOM(ALEA);
```

```
  END;
```

```
  RL(I)=F(C(1),C(2),C(3));
```

```
  IF RMAX < RL(I) THEN RMAX=RL(I);
```

```
END;
```

```
DO I=1 TO N;
```

```
  IF RMAX*RANDOM(ALEA) <= RL(I) THEN NP=NP+1;
```

```
END;
```

```
AREA=1;
```

```
DO I=1 TO 3;
```

```
  AREA=AREA*RJ(I);
```

```
END;
```

```
AREA=AREA*RMAX;
```

```
PUT LIST (NP/N*AREA);
```

```
F:PROCEDURE (V1,V2,V3);  
  RETURN (V1+V2+V3+V1*V2+V1*V3+V2*V3);  
END F;
```

END INTEGRAL;

Es evidente lo compacto que queda la función en *APL* y nótese como no se tuvo necesidad de iterar ni una sola vez. En cambio, en los demás lenguajes si se tuvo que recurrir a la iteración. Esto es debido a la gran ventaja de que *APL* opera con arreglos también. Basic tiene también operaciones con matrices, pero son muy restringidas en comparación con *APL*.

En todos los programas, los datos de entrada fueron:

0,1,0,1,0,1,1000,16807

Pasando a los tiempos de proceso, podemos ver lo siguiente:

Algol, Basic, Fortran y PL/I tienen que pasar por un proceso de compilación, y *APL* no, debido a que es un intérprete (Las instrucciones se interpretan una por una, generándose en ese momento el código, mientras que el compilador genera de una vez todo el código al principio).

Para saber el tiempo de ejecución en *APL*, hay un símbolo que es 'T', el cual si es seguido de determinados enteros a la derecha nos da diferentes informaciones sobre datos del sistema; seguida del número 21 nos da el tiempo de procesador usado en una sesión (en sesentavos de segundo).

Para calcular el tiempo que se lleva ejecutar una función, definimos la siguiente función:

```
VZ←TIEMPO;T  
[1] Z←60 60 60*(T+r21)-TP  
[2] TP←T  
V
```

El número *TP* tiene que estar definido anteriormente.

Llamamos a la función *TIEMPO* antes y después de llamar a la función *INTEGRAL*, y el vector que nos da la segunda vez van a ser los minutos, segundos y centésimas de segundo empleados en el proceso.

La función *APL* fue ejecutada en una computadora IBM 370/155 perteneciente a IBM de México S.A.. Los programas en los demás lenguajes fueron ejecutados en una computadora Burroughs B-6700 perteneciente al Centro de Servicios de Computo de la Universidad Nacional Autónoma de México, todos ellos desde una terminal remota de pantalla.

Ahora sí ejecutemos la función

*TIEMPO*

52 41 47

0 1 0 1 0 1 *INTEGRAL* 1000

2.250310443

*TIEMPO*

0 3 28

El programa de Algol entregó como resultado 2.2530882204, usó de procesador 1.34 segundos en la compilación y 0.99 en la ejecución; el de Basic entregó 2.2530882204 y usó 1.20 en compilación y 1.48 en ejecución; Fortran dio como resultado 2.253882204 consumiendo 1.23 segundos en compilación y 1.23 en ejecución; finalmente PL/I sacó el resultado 2.2530878002 gastando 4.30 segundos en compilación y 1.81 en ejecución. La siguiente tabla ilustra mejor la situación:

Lenguaje	Tiempo de procesador	Tamaño del programa
ALGOL	2.33	27
APL	3.28	3
BASIC	2.68	28
FORTRAN	2.46	24
PL/I	6.11	26

Hay que hacer notar que estos datos referentes a tiempos de proceso pueden ser engañosos en cuanto a la comparación de dos computadoras completamente distintas (el procesador de una puede ser más lento que el de la otra y viceversa).

Aunque es un poco más tardado el *APL* (por ser intérprete) sólo necesita tres instrucciones para realizar lo que otros lenguajes hacen en más de 20.

Es verdaderamente lamentable que en la Facultad de Ciencias y en toda nuestra Universidad en general no se le haya dado difusión al *APL*. No con esto estamos diciendo que *APL* es superior a todos los lenguajes, sino que tiene sus ventajas para determinado tipo de problemas como las tienen otros lenguajes en otro tipo de problemas.

Resumiendo, *APL* es antes que un lenguaje de programación, un lenguaje algebraico no ambiguo que muy probablemente es más comprensible que el lenguaje tradicional del álgebra y debería divulgarse más su uso.



## BIBLIOGRAFIA

APL/360 introducción

IBM corporation.

APL/360 Manual de referencia para el usuario

IBM corporation.

Burroughs B6700/B7700 Algol language reference manual

1974

Burroughs B6700/B7700 APL/700 user reference manual

1975

Burroughs B6700/B7700 Basic language

1971

Burroughs B6700/B7700 Cande Laguage Manual

1973

Burroughs B6700/B7700 Fortran reference manual

1972

Burroughs B6700/B7700 PL/I language manual

1972

Buslenko N.P., Golenko D.I., Shreider Y.A., Sobol I.M., Sragovich V.G.

The Monte Carlo Method, the method of statistical trials

Pergamon

1966

Gilman L., Rose A.J.

*APL* an interactive approach

Wiley

1974

Hammersley J.M., Handscomb D.C.

Monte Carlo methods

Methuen

1964

Iverson K.E.

Elementary Algebra

IBM corporation

1971

McCracken D.D.

The Monte Carlo method

Readings from Scientific American, Computers and Computation

1955

Meyer H.A.

Symposium on Monte Carlo methods

Wiley

1956

Rickmers A.D., Todd H.N.

Introducción a la Estadística

CECSA

1971



**BIBLIOTECA**  
**INSTITUTO DE ECOLOGIA**  
**UNAM**

Sammet J.E.

Programming languages, History and Fundamentals

Prentice Hall

1969

Smillie K.W.

APL/360 with Statistical Applications

Addison Wesley

1974