# Conceptos básicos de programación - II

## 1. Manejo de Vectores y Matrices

Tenemos una tabla como la que mostramos en la Fig. 1 con valores X e Y. Para manejar una serie de datos en forma de vector o matriz en Fortran se definen variables dimensionadas. En el programa **dim1.f** lo primero que hacemos es **definir o asignar** (instrucción **dimension**) dos variables X y Y que tienen un **máximo** de 10 elementos cada una.

Antes de usar este tipo de variable dimensionada, es necesario definirlas al comienzo del programa para reservar espacio en la memoria del ordenador. El programa lee los valores de X y de Y y los guarda en los vectores X(I) e Y(I) (instrucción **read**). Después suma todos los X y todos los Y. Le falta escribir los resultados.

*EDITE, COMPILE Y EJECUTE el programa* **dim1.f**.
*Añada las instrucciones necesarias para obtener la salida de resultados.*
*Estudie con detenimiento el programa, si no entiende algo pregunte a su profesor.*

```
C dim1.f                26 Oct 09
C Autor: .....ps
C Suma los valores de X y de Y
C = = = = = = = = = = = = = = = = = =
      dimension x(10), y(10)
      read*, n

      do i=1,n
          read*, x(i), y(i)
      enddo

      sx = 0.0
      sy = 0.0
      do i=1,n
       sx = sx + x(i)
       sy = sy + y(i)
      enddo

      stop
      end
```

De modo similar se pueden definir matrices, por ejemplo con la instrucción

DIMENSION ARRAY(10,10), B(5,5,5)

definimos una matriz llamada ARRAY de $10 \times 10$ elementos y otra llamada B de $5 \times 5 \times 5$ elementos, sin embargo, esto, como otras muchas instrucciones de Fortran, se sale del límite de esta introducción. Para su estudio le remitimos a cualquier libro de Fortran.

## 2. Ejercicio 2: Ajuste de una recta por mínimos cuadrados

Para ajustar una serie de $N$ puntos $x_i$ e $y_i$ a una recta ($y = a + b \cdot x$) por mínimos cuadrados utilizaremos las ecuaciones siguientes. Primero definimos $S_X, S_Y, S_{XX}$ y $S_{XY}$ como:

$$S_X = \sum_{i=1}^{N} x_i \; ; \quad S_Y = \sum_{i=1}^{n} y_i$$

$$S_{XX} = \sum_{i=1}^{N} x_i^2 \; ; \quad S_{XY} = \sum_{i=1}^{n} x_i y_i$$

| $x_i$ | $y_i$ |
|---|---|
| 1. | 3.7 |
| 2. | 7.4 |
| 3. | 9.3 |
| 4. | 13.2 |
| 5. | 15.9 |
| 6. | 19.1 |
| 7. | 20.7 |
| 8. | 25.5 |
| 9. | 27.6 |

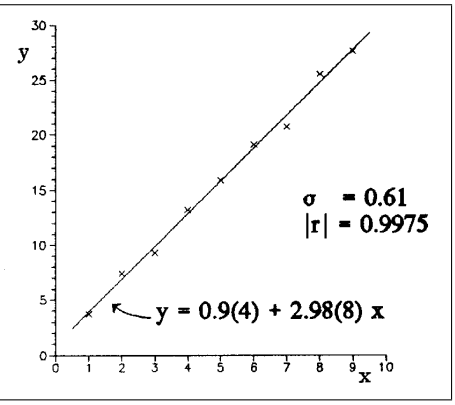


Figura 1: Recta ajusta por mínimos cuadrados.

Con estas definiciones, el corte en ordenadas $(a)$ y la pendiente $(b)$ de la recta $(y = a + b \cdot x)$ vienen dados por:

$$a = \frac{S_Y - b \cdot S_X}{N} \; ; \quad b = \frac{S_X \cdot S_Y - N \cdot S_{XY}}{S_X \cdot S_X - N \cdot S_{XX}}$$

La desviación cuadrática media del ajuste se define como:

$$\begin{aligned} s_y &= \sqrt{\frac{\sum_{i=1}^{N}(y_i - f(x_i))^2}{N-2}} \\ &= \sqrt{\frac{\sum_{i=1}^{N}(y_i - (a+bx_i))^2}{N-2}} \\ &= \sqrt{\frac{SD2}{N-2}} \; ; \end{aligned}$$

con

$$SD2 = \sum_{i=1}^{N}(y_i - (a+bx_i))^2$$

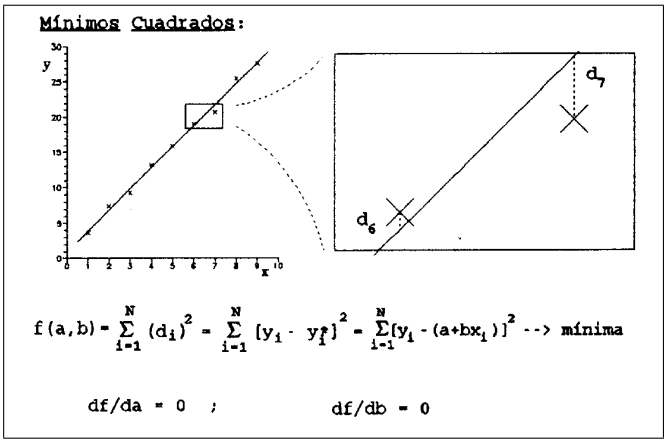


Figura 2: Un ajuste de mínimos cuadrados se basa en minimizar la suma del cuadrado de la diferencias entre los puntos que usamos en el ajuste $(y_i)$ y los puntos de la recta ajustada $(y_i^* = f(x_i))$.

y las desviaciones estándar del corte en ordenadas $(s_a)$ y de la pendiente $(s_b)$ se estiman como:

$$s_a = s_y \sqrt{\frac{S_{XX}}{N \cdot S_{XX} - S_X \cdot S_X}} \; ; \qquad s_b = s_y \sqrt{\frac{N}{N \cdot S_{XX} - S_X \cdot S_X}}$$

El coeficiente de correlación se puede calcular con la expresión siguiente:

$$r = \frac{S_X \cdot S_Y - N \cdot S_{XY}}{\left[(S_X^2 - N \cdot S_{XX}) \cdot (S_Y^2 - N \cdot S_{YY})\right]^{1/2}} \qquad \text{con} \quad S_{YY} = \sum_{i=1}^{N} y_i^2$$

**[P]** *Utilizando las ecuaciones anteriores haga un programa que lea N puntos $x_i$ e $y_i$ y ajuste una recta por mínimos cuadrados.*

En la Fig. 1 se dan los resultados para este ejemplo. Las incertidumbres de la pendiente y del corte en ordenadas correspondientes a la última cifra significativa se dan entre paréntesis.

# 3. Subprogramas

La realización de programas complejos se suele hacer dividiendo el programa en trozos, cada uno de los cuales realiza una tarea definida. Estos trozos es lo que llamaremos subprogramas. A veces hay un conjunto de instrucciones que se repiten en un mismo programa. Si programamos estas instrucciones en un subprograma, el programa principal únicamente tiene que «llamar» a dicho subprograma sin repetir las instrucciones. En realidad, las funciones intrínsecas que hemos visto anteriormente son subprogramas realizados por los desarrolladores del compilador.

Subprogramas y funciones

input ⟶ subprograma ⟶ output

- Conjunto de instrucciones repetitivas.
- Instrucciones que por su generalidad pueden usarse en otros programas.
- Ahorra tiempo de programación, memoria en el ordenador y mejora la estructura de los programas.
- Dos tipos de subprogramas:
  - subroutine
  - function

Figura 3: Subprogramas y funciones.

La programación con subprogramas facilita el trabajo del programador dando lugar a una programación más estructurada. En Fortran hay dos tipos principales de subprogramas:

1. Subrutina, **subroutine**. Cuando la tarea a realizar modifica varias variables aunque también se puede usar cuando modifica una sola variable o ninguna.
2. Función, **function**. Cuando la tarea a realizar devuelve un único valor escalar.

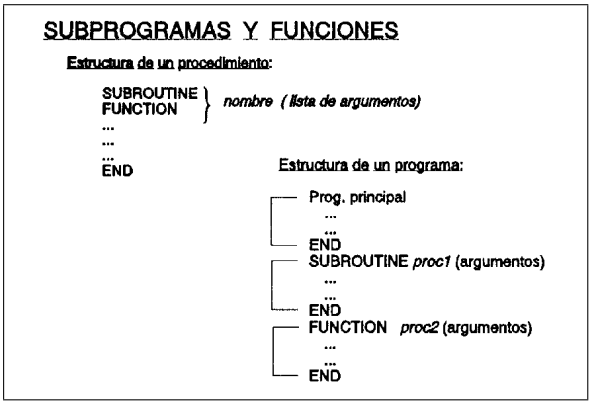


Figura 4: Estructura de un subprograma y de un programa.

Estos subprogramas realizan tareas independientes y pueden compartir valores (argumentos) con el programa que les «llama».

Como puedes ver en los ejemplos que mostramos a continuación, el subprograma se escribe después del programa principal, aunque puede escribirse antes e incluso en un fichero aparte pero **nunca** puede ir intercalado entre las instrucciones del programa principal (main en ingles).

En el caso de la subrutina, se las llama con la instrucción **call** seguida del nombre de la subrutina y entre paréntesis se escriben las variables o constantes (argumentos) que va a utilizar el subprograma.

La llamada a una función se realiza del mismo modo que con las funciones intrínsecas (observa las diferencias entre **suma1.f** y **suma2.f**). Cuando se efectúa la llamada a un subprograma, el programa principal cede el control al subprograma, éste realiza su tarea y al finalizarla devuelve el control al programa principal en la instrucción siguiente al **call**. Las variables modificadas en los subprogramas **y** que están compartidas con el programa principal quedan también modificadas en el programa principal. Hasta ahora hemos hablado de programa principal y

subprograma, sin embargo, un subprograma puede a su vez llamar a otros subprogramas, siempre que la llamada **no sea recursiva**, es decir un subprograma no puede llamarse a sí mismo.

```
C suma1.f        17 Oct 97
C Subrutina suma 2 numeros
C = = = = = = = = = = = = =
C Lectura de datos ......
   READ*, A, B
   CALL SUMA(A,B,C)
   PRINT*, A, B, C
   CALL SUMA(A,C,D)
   PRINT*, A,C,D
   STOP
   END
C ========================
C Subrutina SUMA:
C    R3 = R1 + R2
C ========================
  SUBROUTINE SUMA(R1,R2,R3)
  R3 = R1 + R2
  RETURN
  END
```

```
C  suma2.f      17 Oct 97
C  Funcion suma 2 numeros
C = = = = = = = = = = = = =
C Lectura de datos ....
      READ*, A, B
      C = SUMA(A,B)
      PRINT*, A, B, C
      D = SUMA(A,C)
      PRINT*, A,C,D
      STOP
      END
C=========================
C FUNCION SUMA:
C    SUMA = R1 + R2
C=========================
     FUNCTION SUMA(R1,R2)
     SUMA = R1 + R2
     RETURN
     END
```

```
C media1.f      17 Oct 97
C Calcula la media de los
C datos de un vector
C = = = = = = = = = = = = =
        DIMENSION V(100)
C Lectura de datos ......
        READ*, N
        DO I=1,N
        READ*, V(I)
        ENDDO
C Llamo a la subrutina MEDIA
    CALL MEDIA (V,N,RMEDIA)
    PRINT*, RMEDIA
    STOP
    END
C  ==========================
C Subrutina MEDIA:
C    Z es un vector
C    M es el numero de
C      elementos del vector
C    RM es la media de los
C      elementos
C ==========================
    SUBROUTINE MEDIA(Z,M,RM)
    DIMENSION Z(100)
    SUM = 0.0
    DO I=1,M
      SUM = SUM + Z(I)
    ENDDO
    RM = SUM / M
    RETURN
    END
```

Un subprograma comienza con las instrucciones **subroutine** o **function** seguido del nombre y entre paréntesis los argumentos (ver Fig. 4). Los argumentos tienen que ser los mismos en la llamada del programa principal y en el subprograma y, por tanto, aunque pueden cambiar de nombre no pueden cambiar de tipo (reales, enteros, caracteres, ...) ni de orden, es decir en el programa **suma1.f** A, B y C en la primera llamada a la subroutina son en la subroutina R1, R2 y R3, respectivamente.

Las variables dimensionadas hay que volver a definirlas en los subprogramas (DIMENSION). Los subprogramas terminan con la instrucción **return** que devuelve el control al programa principal (o al subprograma que llamó a la subrutina).

> Edite, compile y ejecute los programas **suma1.f, suma2.f y media1.f**. Estudie cuidadosamente como funcionan y las diferencias que existen entre ellos.

Hay otra forma de compartir argumentos entre el programa principal y los subprogramas y viceversa, es la instrucción **common**. La declaración **common** sitúa las variables en una posición de memoria especial a la que tienen acceso las subrutina que contengan el mismo **common**. Cada COMMON puede llevar asociado un nombre, aunque existe el llamado «black» **common** que no tiene nombre. Por ejemplo:

```
COMMON/NOTAS/ JUN(25), SEP(5)
```

es un **common** de nombre NOTAS que guarda las variables JUN(25) y SEP(25) en una posición de memoria a la que pueden acceder las subrutina que contengan el common/NOTAS/ ...
Un ejemplo de «black» **common** es:

```
common ARRAY(225,2), VEC(7), RR
```

# 4. Ejercicio 3: Integración numérica

El valor de una integral definida

$$\int_a^b f(x)dx$$

viene dado por el área limitada por la curva $y = f(x)$ y el eje de abscisas ($x$) entre $x = a$ y $x = b$ (Fig. 5).

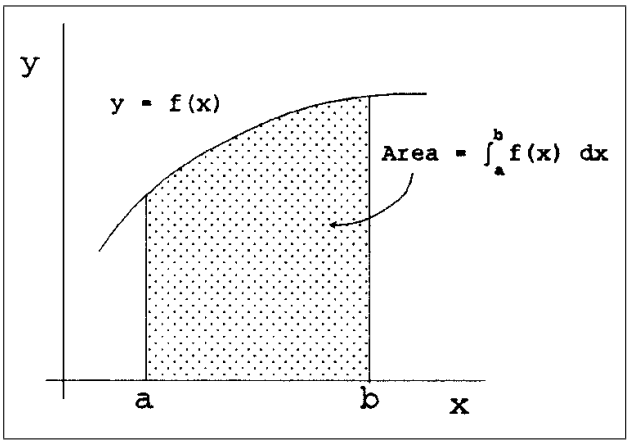


Figura 5: Integral de una función.

Se puede realizar una estimación del área dividiendo la superficie total en pequeñas porciones y sumando el área de cada porción, tal y como se muestra en la Fig. 6. En el método trapezoidal, la superficie de cada porción se aproxima a un trapecio y por lo tanto su área viene dada por la ecuación

$$A_i = \frac{h}{2}[f(x_i) + f(x_{i-1})]$$

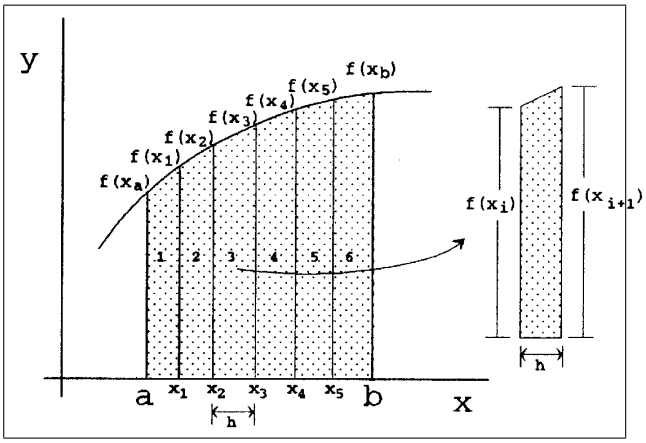


Figura 6: Integral o área como suma del área de los trapecios

Sumando el área de todas las porciones o trapecios, en este caso para un ejemplo con N (n° de trapecios) igual a 6, tenemos:

$$\begin{aligned} I_T &= \frac{h}{2}[f(a)+f(x_1)] + \frac{h}{2}[f(x_1)+f(x_2)] \\ &+ \frac{h}{2}[f(x_2)+f(x_3)] + \frac{h}{2}[f(x_3)+f(x_4)] \\ &+ \frac{h}{2}[f(x_4)+f(x_5)] + \frac{h}{2}[f(x_5)+f(b)] \end{aligned}$$

que podemos reescribir en general como:

$$I_T = \int_a^b f(x)\,dx = \frac{h}{2}\left[f(a) + 2.0\sum_{i=1}^{N-1} f(x_i) + f(b)\right] \quad (1)$$

**[P]** *Escriba un programa Fortran para integrar numéricamente por el método de los trapecios. Aplique el programa al cálculo de la siguiente integral:*

$$I_T = \int_0^2 (6x^5 - 7)\,dx$$

Siga el siguiente esquema:

1. Organice la lectura de datos ($a, b$ y $N$).

2. Calcule $h$ como $(b-a)/N$.
3. Calcule los valores de $f(a)$ y $f(b)$ llamando a la función (ver punto 6).
4. Calcule la parte correspondiente al sumatorio. $SUM = \sum_{i=1}^{N-1} f(x_i)$ con $x_i = a + i \cdot h$.
5. Aplique la Ec. (1) y escriba los resultados.
6. Programe la función $f(x) = 6x^5 - 7$ como un subprograma de tipo FUNCTION.

# Referencias


[1] `http://en.wikipedia.org/wiki/Fortran-`

[2] Pedro Alberto Enriquez Palma y M$^a$ Pilar Puyuelo García, *Introducción a la programación en Fortran 90 para Químicos. Prácticas*, Universidad de la Rioja, Servicio de Publicaciones, (2006). `http://www.unirioja.es/servicios/sp/catalogo/online/fortran90.pdf`.

[3] .Elementos de PROGRAMACIÓN FORTRAN para CIENTÍFICOS e INGENIEROS"de Manuel Fernández, Rosa Rodríguez, David Zorrilla y Jesús Sánchez. UCA (Editorial: Real Sociedad Española de Física, C.T.P.M 2006).

[4] Introducción al Fortran 90: `http://www.cesga.es/telecursos/F90/`.

[5] Curso breve de Fortran: `http://www.uam.es/departamentos/ciencias/fisicateoricamateria/especifica/hojas/kike/FORTRAN/FORTRAN90.html`.