1.6 Manejo de ficheros-m:

1.6.1 Programas y funciones

- Un fichero-m es un fichero de texto ASCII con la extensión '.m' que contiene comandos de MATLAB. Dos tipos:

  - **Programas**: No acepta argumentos de entrada o salida explícitos. Los aporta.

    Llama a una función (o a otro programa).

  - **Funciones**: Comienzan con una línea con el comando 'function' y pueden aceptar argumentos de entrada y producir argumentos de salida

Ejemplo 1: Codificación de una función que devuelve el signo de una variable entera n.

-Si n es estrictamente negativa el signo se define como -1,

-si n es 0 el signo será 0,

-y si n es estrictamente positiva, el signo se define como +1.

-Hay que gestionar tres posibles excluyentes; se aborda una, eliminándola frente a las otras dos y después las otras dos entre si.

```
n=round((rand-0.3)*10)-4
if n<0
signo=-1;
elseif  n>0
signo=1;
else
signo=0;
end
signo
```

Programa: callsignobase

```
n=round((rand-0.3)*10)-4
signo= fsignobase(n)
```

Función: fsignobase

```
function signo= fsignobase(n)
n=round((rand-0.3)*10)-4
if n<0
signo=-1;
elseif   n>0
signo=1;
else
signo=0;
end
signo
```

Programa: callsignobase

```
n=round((rand-0.3)*10)-4
signo= fsignobase(n)
```

Función: fsignobase

```
function signo= fsignobase(n)
if n<0
signo=-1;
elseif   n>0
signo=1;
else
signo=0;
end
signo
```

PROGRAMA

```
a=3
b=5
c=8
y=pro(a,b,c)
```

FUNCION

```
function y=pro(a,b,c)
if a>b & c>b
    y=a+b+c;
    z=y-b;
else
    if a>b | c>b
        y=a+b+c;
        z=y-c;
    else
        y=a-b-c;
        z=y-a;
    end
end
z
```

1.6.1 Programas y funciones

- Ejemplo 2: función con operadores elemento a elemento. Cálculo del polinomio de Chebyshev de grado p en el vector $[x_1 \cdots x_n]$

$T_0(x) = 1$ (primer línea)

$T_1(x) = x$ (segunda línea)

$T_p(x) = 2x \cdot T_{p-1}(x) - T_{p-2}(x)$ $\quad p \geq 2$ (tercera línea y siguientes)

El cálculo del polinomio de Chebyshev va a consistir en crear una **matriz** con los resultados de ese cálculo.

¿Cuantas filas va a tener la matriz? Tantas como expresiones de T tenga, es decir tantas como el valor del grado del polinomio+1. Ejemplo: Si llegamos a p=10, tendremos 11 filas. Si llegamos a p=28 tendremos 29 filas. k es el número de filas en la matriz.

$$k = p + 1$$

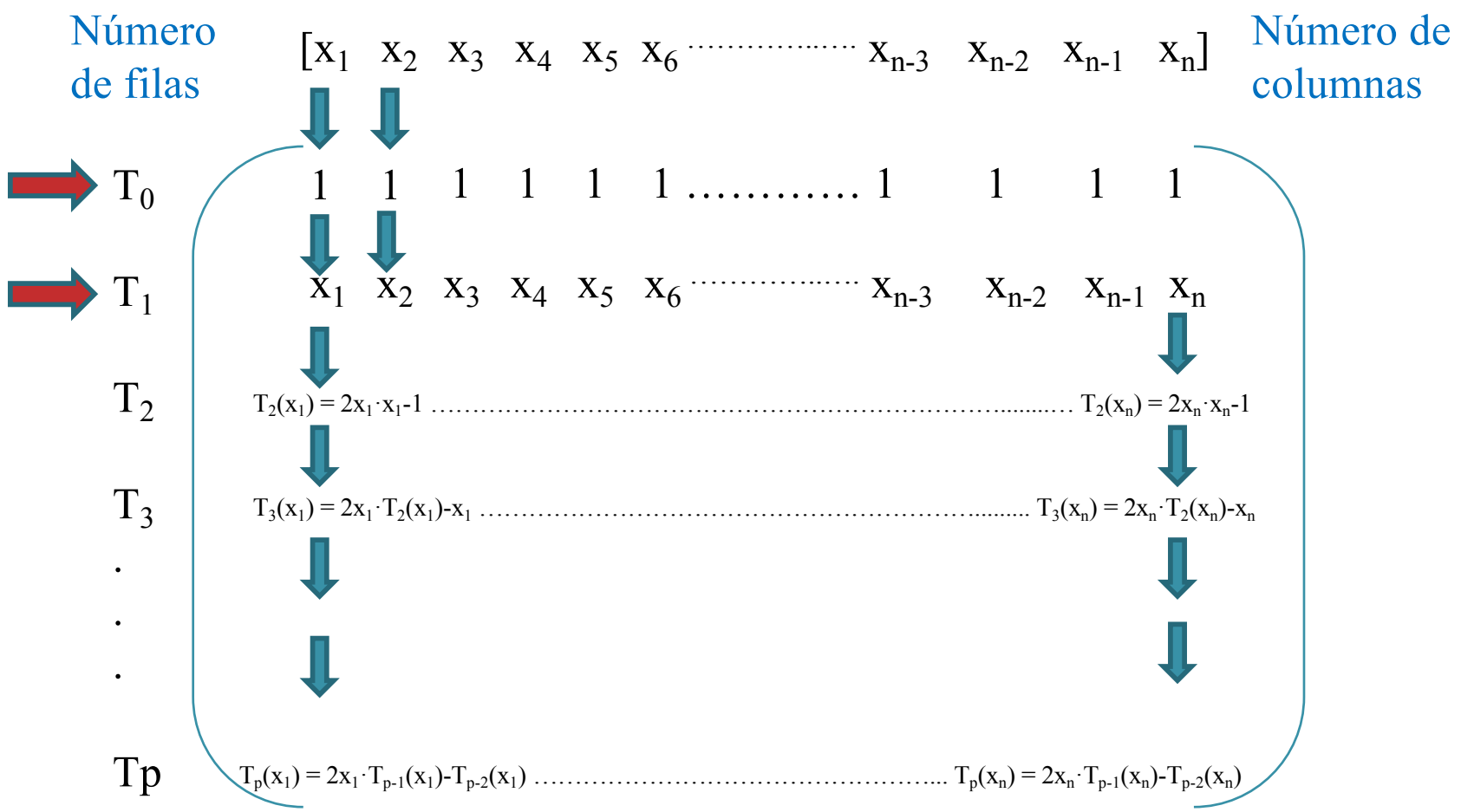
	k
T_0	1
T_1	2
T_2	3
T_3	4
.	.
.	.
.	
Tp	P+1

¿Cuantas columnas va a tener la matriz? Tantas como valores de la variable de x tenga, es decir tantas como valores del vector x tenga.

$$x = [x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \cdots\cdots\cdots\cdots\cdots\cdots \quad x_{n-3} \quad x_{n-2} \quad x_{n-1} \quad x_n]$$

$T_p(x) = 2x \cdot T_{p-1}(x) - T_{p-2}(x)$

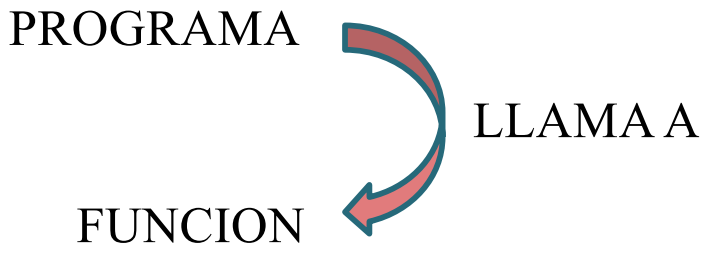
1.6.1 Programas y funciones



Por tanto, tenemos que crear en matlab una matriz de k filas (k=p + 1) y n columnas para mostrar los cálculos del polinomio.

1.6.1 Programas y funciones

Vamos a crear dos archivos:



¿Qué debe tener el programa?

- En el programa vamos a proporcionar el vector de valores de x sobre el que voy a aplicar y calcular el polinomio de Chebyshev.

  Va a ser un vector fila de valores de x

  Por ejemplo: x= [1:1:10]

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

1.6.1 Programas y funciones

- Además, voy a hacer que el programa proporcione un valor de grado de polinomio p de manera aleatoria pero que pueda ser relativamente grande.

Creo una variable m que se calcula de forma aleatoria, lo redondeo y lo multiplico por 10 para que sea un valor mayor que 1 o 2.

```
m=10*randn;
'el grado del polinomio a calcular es'
p=round(abs(m))
```

- Tenemos que incluir la llamada del programa a la función y también voy a pedir que haga la representación gráfica

```
y=fcheby(x,p)
plot(x,y)
```

1.6.1 Programas y funciones

El programa quedaría así:

```
x=[1:1:10];
m=10*randn;
'el grado del polinomio a
calcular es'
p=round(abs(m))
y=fcheby(x,p)
plot(x,y)
```

Este archivo lo haríamos en el editor y lo grabaríamos con el nombre que creamos conveniente, por ejemplo, lo podemos llamar cheby.

Tiene este programa la llamada al archivo de la función. Dicha función tendrá que llamarse obligatoriamente fcheby.

1.6.1 Programas y funciones

FUNCION

El primer paso para hacer la función es escribir en la primera línea del editor la palabra function y seguidamente y= nombre de la función (variables de las que depende la función)

```
function y=fcheby(x,p);
```

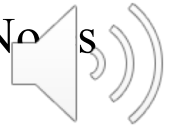
Nota: En este caso el nombre de la función es fcheby, y el archivo de la función será nombrado y grabado en el editor obligatoriamente con este nombre.

Seguidamente, creamos una matriz de unos que se va a llamar y.

```
y=ones(p+1,length(x));
%Inicialización de y de acuerdo con el tamaño de x
```

Inicialmente esta matriz no contiene más que unos pero a medida que vamos calculando los valores de los polinomios de Chebyshev, se sustituyen los unos por los valores calculados. Las dimensiones de la matriz son evidentes: el número de filas son el grado del polinomio al que queremos o debemos llegar más 1, las columnas son el número de valores del vector x, es decir, la longitud de ese vector.

Nota: Recuerdo que cuando se escribe % sirve para poner un comentario. No es necesario que lo pongáis vosotros en un ejercicio.



1.6.1 Programas y funciones

Seguidamente escribimos la primera y segunda filas de la matriz que estamos creando. Sabemos que son valores determinados, no son cálculos.

```
y(1,:)=1;        % Para la matriz y, fila 1, todas las columnas
y(2,:)=x;        % Para la matriz y, fila 2, todas las columnas
```

La siguiente parte sirve para que, si no hemos conseguido un valor de grado de polinomio no más alto de 1, queremos que nos avise Matlab:

```
%Si p es <=1, ya no hacen falta mas calculos%
if p <=1,
    return
end
```

1.6.1 Programas y funciones

Por último, añadimos un ciclo for para calcular las siguientes filas de la matriz, desde la fila 3 hasta la p+1

```
%ciclo que calcula los vectores y(3→p+1,:)
for k=3:p+1
    y(k,:)=2*x.*y(k-1,:)-y(k-2,:);
end
```

De esta forma calculo los polinomios que dependen del anterior y del dos veces anterior.

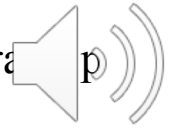
Además, le pido que en vez de enseñarme toda la matriz, sólo me enseñe la última fila de ésta.

```
y=y(p+1,:)
 %Sólo necesitamos el polinomio de grado p%
```

1.6.1 Programas y funciones

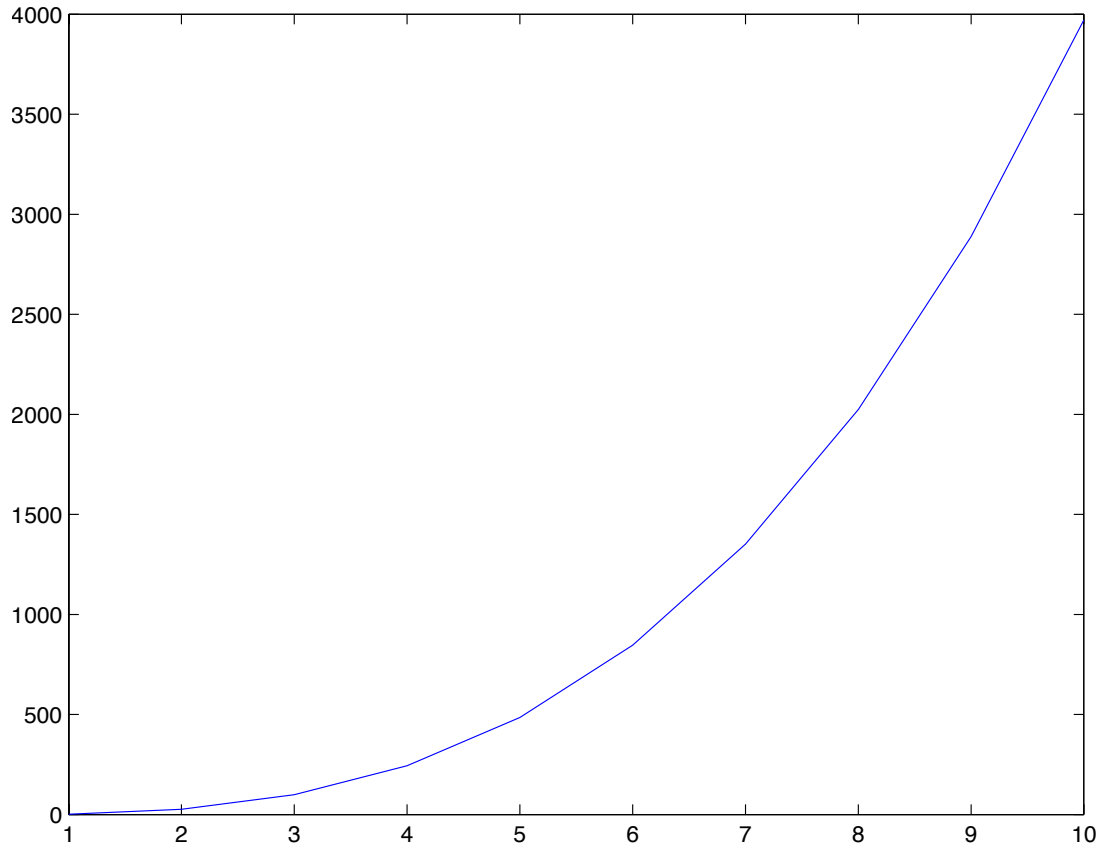
```
x=[1:1:10];
m=10*randn;
'el grado del polinomio a
calcular es'
p=round(abs(m))
y=fcheby(x,p)
plot(x,y)
```

```
function y=fcheby(x,p);

y=ones(p+1,length(x));
%Inicialización de y de acuerdo con el tamaño de x
    y(1,:)=1; %No es necesario

    y(2,:)=x;

if p <=1, return, end %Si p es <=1, ya no hacen falta
más cálculos%

%ciclo que calcula los vectores y(3→p+1,:)

for k=3:p+1
   y(k,:)=2*x.*y(k-1,:)-y(k-2,:);
end
y=y(p+1,:); %Sólo necesitamos el polinomio de gra   p
```



1.6.1 Programas y funciones

PLOT DEL PROGRAMA CHEBY



EJEMPLO POLINOMIOS SUCESIVOS:

Crear una función que calcule el polinomio de grado *p* en el vector $[x_1 \ldots x_n]$ que tiene la siguiente fórmula de recurrencia:

$P_{-1}(x) = 1,3$

$P_0(x) = 5,2x$

$P_1(x) = 6x+2$

$P_2(x) = x^2+x+3$

$P_i(x) = 6x\,P_{i-1}(x) - i \cdot P_{i-2}(x)$ para $i > 3$

Una vez creada, hacer un programa principal donde se crea un vector de 800 elementos en el intervalo [0,1], se llama a la función para que muestre el polinomio de grado 11, y se representa gráficamente P(x) en dicho intervalo.

EJEMPLO POLINOMIOS SUCESIVOS:

Crear una función que calcule el polinomio de grado *p* en el vector $[x_1 \dots x_n]$ que tiene la siguiente fórmula de recurrencia:

$P_{-1}(x) = 1,3$

$P_0(x) = 5,2x$

$P_1(x) = 6x+2$

$P_2(x) = x^2+x+3$

$P_i(x) = 6x\,P_{i-1}(x) - i \cdot P_{i-2}(x)$ para $i > 3$

Una vez creada, hacer un programa principal donde se crea un vector de 800 elementos en el intervalo [0,1], se llama a la función para que muestre el polinomio de grado 11, y se representa gráficamente P(x) en dicho intervalo.

PROGRAMA

```
x=linspace(0,1,800);
m=20*randn;
'el grado del polinomio a calcular es'
p=round(abs(m))
y=fsem9c1516GA(x,p)
plot(x,y)
```

FUNCION

```
function y=fsem9c1516GA(x,p)
 %inicializamos la matriz%
y=ones(p+2,length(x));
 %calculamos los terminos iniciales%
y(1,:)=1.3;
y(2,:)=(5.2).*x;
y(3,:)=6.*x+2;
y(4,:)=x.^2+x+3;
%calculamos el resto de los terminos de la matriz%
for j=5:p+2
i=j-2;
    y(j,:)=6.*x.*y(j-1,:)-i.*y(j-2,:);
end
y=y(13,:);
```

COMENTARIOS:

y(j,:)=6.*x.*y(j-1,:)-i.*y(j-2,:);

No confundir la nomenclatura para utilizar linspace a la nomenclatura para crear un vector directamente

x=linspace(0,1,800)

Con linspace creo un vector fila que va del valor inicial 0 al valor final 1 y tendrá 800 elementos

x= [1:1:10]

Creamos un vector que va de 1 a 10 en pasos de 1

DISCRIMINAR NUMEROS PARES E IMPARES:

- La comparación que debemos hacer para saber si una VARIABLE n es par o impar es n/2==fix(n/2)

Ej: n=8
8/2=4 fix(8/2)=4
n=9
9/2=4,5 fix(9/2)=4

```
n=round(10*rand)
if(n/2==fix(n/2))
tipo= 'par'
else
tipo= 'impar'
end
```

- rem: es una aplicación de fix. Puede emplearse para escalares, vectores o matrices. Determina si el elemento es par o impar escribiendo 0 o 1. La operación que hace es la siguiente:

r= a – b * fix(a/b)

```
a= round(10*rand(1,10))
b = 2;
r= rem(a,b)
```

EJEMPLO:

```
a =

    0    9    7    4    7    1    2    8    2    8

r =

    0    1    1    0    1    1    0    0    0    0
```