Polinomios

Un Polinomio es una ecuación matemática que se expresa como la suma de términos finitos que contienen un coeficiente, una variable en común y un exponente diferente en cada uno de ellos, es decir, un polinomio se puede expresar de la siguiente forma:

$$P(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots + a_nx^n$$

En este caso, a_i son los coeficientes, x es la variable en común y podemos ver que los exponentes toman valores desde 0 hasta n.

Es importante también considerar que se llama grado del polinomio al exponente más grande que éste tiene. Por ejemplo, en el polinomio $P(x) = 3x^2 + 5x - 2$ el exponente más grande que tiene es 2 por lo tanto se dice que es de grado 2, también se le puede decir que este polinomio es de segundo grado o de segundo orden.

En Matlab podemos manejar polinomios de una manera directa sin embargo es necesario tener las siguientes consideraciones:

- En Matlab se escriben únicamente los coeficientes del polinomio (incluyendo su signo) como si fueran elemento dentro de un vector fila (se deben escribir entre corchetes separados por comas o espacios)
- Para escribir los coeficientes primero hay que ordenarlos del que tiene la variable con exponente mayor hasta el exponente menor
- Si dentro del polinomio que queremos escribir no existe un término con algún exponente determinado, entonces el coeficiente que tenemos que escribir en Matlab será 0, no se podrá saltar.
- Un vector que simboliza un polinomio de grado n siempre tiene longitud n+1
- Al utilizar esta forma de escribir los polinomios en Matlab podemos utilizar comandos especiales para polinomios (roots, polyval, polyfit, entre otras)

Ejemplo:

1.- Escribir el siguiente polinomio en Matlab:

$$P(x) = 3x^2 + 5x - 2$$

Solución

Aquí vemos que los exponentes de los términos ya están arreglados de mayor a menor, por lo tanto, ahora sólo procedemos a escribir dentro de un vector fila los coeficientes de cada término.

P = [3, 5, -2]

P en este caso es un vector fila de 3 elementos que representa al polinomio grado 2 del problema, por lo tanto, vemos que sí se cumple la regla de que el Polinomio en Matlab tienen longitud ***n + 1***, donde ***n*** es el grado del polinomio.

Es importante además notar que, el último término tiene signo negativo porque en el polinomio el 2 está restando

2.- Escribir el siguiente polinomio en Matlab:

$$P(x) = 4x^2 - 2 + 3x^3 - 10x$$

Solución

Aquí vemos un polinomio con sus términos desordenados, por lo que tendremos que ordenarlos con los exponentes de mayor a menor, quedando de la siguiente forma:

$$P(x) = 3x^2 + 4x^2 - 10x - 2$$

Una vez ordenados entonces sí podemos escribirlos en Matlab como un vector, de la siguiente forma:

P = [3, 4, -10, -2]

En este caso el polinomio que queremos escribir es de grado 3 y la longitud del vector si cumple que sea de ***n+1*** elementos.

3.- Escribir el siguiente polinomio en Matlab:

$$P(x) = 7x^4 - 2x + 12x^3$$

Solución

Podemos ver que este es un polinomio de grado 4 y está desordenado, por lo que tenemos que ordenarlo, quedando de la siguiente manera:

$$P(x) = 7x^4 + 12x^2 - 2$$

En este caso, podemos también observar que hacen falta los términos que tienen los exponentes 3 y 1, por lo tanto, podemos completar los coeficientes con ceros:

$$P(x) = 7x^4 + 0x^3 + 12x^2 + 0x - 2$$

Ya que realizamos este procedimiento, podemos escribir en Matlab el polinomio de la siguiente forma:

```
P = [7, 0, 12, 0, -2]
```

Al realizar esto podemos seguir comprobando que el vector que escribimos tendrá ***n+1*** elementos.

Comandos para polinomios

- **roots (raíces)**

Las raíces de cualquier función matemática ***f(x)*** se refiere al valor que tiene ***x*** cuando ***f(x)=0***. Geométricamente esto se vería como el punto en el cual la gráfica de ***f(x)*** cruza con el eje de las abscisas (eje x).

En un polinomio grado ***n*** se sabe que tiene al menos ***n*** raíces. En algunos casos estas ***n*** raíces se pueden observar en la gráfica del polinomio, siempre y cuando se cumpla que todas son reales.

Por ejemplo, podemos calcular las raíces de un polinomio grado 2 de la forma:

$$P(x) = ax^2 + bx + c$$

Utilizando la fórmula general:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Sabemos que la expresión anterior nos da 2 resultados (uno para la suma y otro para la resta), estos son las raíces de dicho polinomio y que podemos ver gráficamente siempre y cuando sean reales, es decir los valores de ***a***, ***b*** y ***c*** sean tales que el radicando no sea negativo, si éste sale negativo las raíces entonces serán complejas (imaginarias).

La función roots de Matlab nos permite calcular las raíces de un polinomio guardado con la forma que se vio anteriormente, su sintaxis es muy sencilla solamente se escribe

```
roots(P)
```

El resultado que este comando arrojará será un vector columna de ***n*** elementos que contiene las raíces del polinomio ***P***.

Ejemplo:

4.- Calcular las raíces del siguiente polinomio

$$P(x) = -2x^2 + 2x + 1$$

Solución

Tenemos que escribir el polinomio en el mismo formato que se vio antes, en este caso el polinomio no está desordenado y se tienen todos los coeficientes por lo tanto quedaría de la siguiente manera:

```
P = [-2, 2, 1]
roots(P)
```

El resultado que arroja el programa es:

```
 1.3660
-0.3660
```

Sustituyendo los valores de a, b y c del polinomio en la fórmula general entonces nos debería de dar los datos anteriores.

5.- Calcular las raíces del siguiente polinomio

$$P(x) = 7x^5 - 3x^2 - x^3 + 12x + 1$$

Solución

Calcular raíces de polinomios de orden superior a 2 analíticamente, se vuelve una tarea bastante compleja, es una de las razones por las que esto es de gran utilidad ya que dejamos los cálculos a la computadora y nosotros únicamente nos preocupamos por escribir bien el polinomio en Matlab.

En este caso vemos que es un polinomio grado 5, que faltan algunos términos y están desordenados, por lo que tenemos que ordenarlos quedando de la siguiente forma:

$$P(x) = 7x^5 + 0x^4 - x^3 - 3x^2 + 12x + 1$$

Entonces en Matlab escribimos:

```
P = [7, 0, -1, -3, 12 1]
roots(P)
```

El resultado que arroja el programa es:

```
  0.8554 + 0.7080i
  0.8554 - 0.7080i
 -0.8146 + 0.8686i
 -0.8146 - 0.8686i
 -0.0817 + 0.0000i
```

En este caso todas las raíces fueron complejas.

- polyval (evaluando en polinomios)

Muchas veces cuando tenemos una función matemática ***f(x)***, queremos saber cuánto vale esta si tenemos un valor o valores de ***x*** determinados, para esto analíticamente sustituimos en la expresión el valor que queramos en cada una de las ***x*** que hay dentro de la función.

Por ejemplo, si tenemos

$$f(x) = 3x + 1$$

y queremos saber cuanto vale ***f(x)*** cuando ***x=5***, es decir, ***f(5)***, entonces:

$$f(5) = 3(5) + 1$$

$$f(5) = 16$$

Esto es igualmente posible de realizar en un polinomio escrito en Matlab únicamente utilizando la función polyval, que tiene la siguiente sintaxis:

```
polyval(P,a)
```

Donde: ***P*** es el vector del polinomio que tenemos y ***a*** es el valor que tenemos que sustituir.

Ejemplo:

6.- Suponga que se tienen el siguiente polinomio. Calcula ¿Cuál es el valor de *P(x)* si se sabe que *x = 10*?

$$P(x) = 3x^2 - 2x$$

Solución

Aquí únicamente completamos el coeficiente del polinomio que hace falta de la siguiente forma:

$$P(x) = 3x^2 - 2x + 0$$

Entonces, podemos utilizar el comando polyval con la sintaxis mencionada anteriormente considerando que en este caso el valor de a es de 10 ya que es lo que pide el problema:

```
P = [3, -2, 0];
polyval(P,10)
```

El resultado obtenido es:

```
280
```

- **polyval (gráficas)**

El comando polyval también resulta muy útil si queremos graficar un polinomio. Recordando que para graficar cualquier función matemática primeramente debemos contar con un conjunto de pares ordenados ***x*** y ***y***. El primer conjunto ***x*** lo podemos obtener a partir del intervalo de la gráfica que queremos hacer y el conjunto ***y*** lo obtenemos al evaluar estos valores de ***x*** en la función a graficar.

En Matlab para graficar tenemos que definir los valores ***x*** como un vector y podemos hacer uso de la sintaxis para la creación de vectores con valores linealmente espaciados:

```
x = inicio : incremento : fin
```

Donde: inicio y fin están definidos por el intervalo de la gráfica y el incremento debe de ser suficientemente pequeño para poder tener curvas bien definidas pero que también permita que la ejecución del programa no sea lenta.

Una vez escrito el vector de ***x***, necesitamos el conjunto de valores en ***y*** el cual lo podemos obtener al utilizar el comando polyval para poder evaluar el polinomio ***P***, en los valores que tiene ***x***, quedando:

```
y = polyval(P, x)
```

Finalmente para realizar la gráfica utilizamos el comando **plot(x,y)**.

Ejemplo:

7.- Graficar el siguiente polinomio en el intervalo [-3, 3].

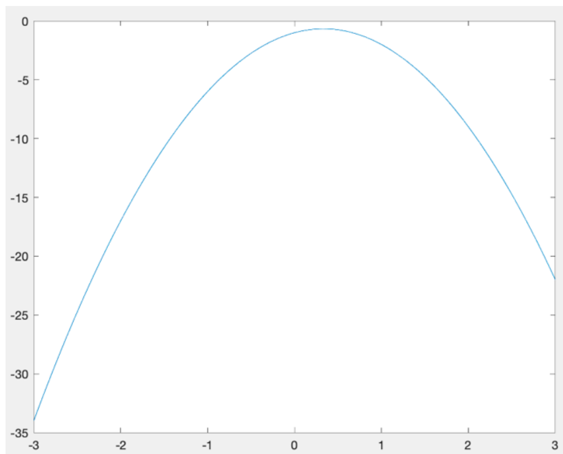
$$P(x) = -3x^2 + 2x - 1$$

Solución

En este caso el polinomio tiene los coeficientes completos y ordenados por lo que realizamos la codificación de la gráfica considerando el incremento de x de 0.01

```
P = [-3, 2, -1];
x= -3 : 0.01: 3;
y = polyval(P, x);
plot(x,y)
```

En este caso no nos interesa ver los valores de ***x y***, ya que no tiene sentido observarlos porque serían muchos valores que no aportarían nada a lo que solicita el problema; por lo que ponemos el carácter punto y coma (**;**) al final de cada sentencia para que no se desplieguen en el programa. Sin embargo, si es importante observar la gráfica obtenida:



En la figura anterior podemos observar que el gráfico obtenido es una parábola lo que se esperaría ya que el polinomio que graficamos es de grado 2.

Ejemplo:

8.- Graficar el polinomio que se muestra a continuación en el intervalo [-2, 1.5] y además dentro del mismo gráfico marcar las raíces del polinomio con asteriscos rojos.

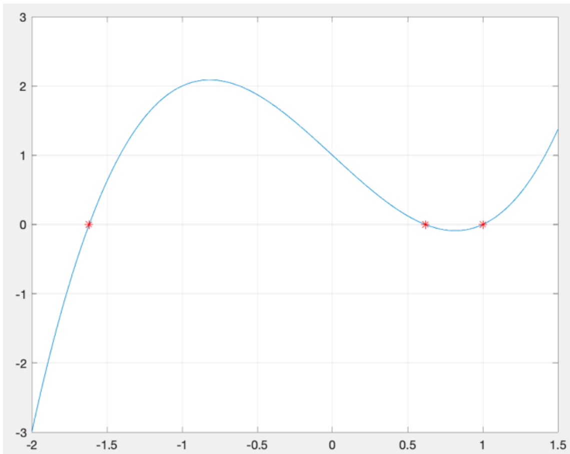
$$P(x) = x^3 - 2x + 1$$

Solución

Este problema ya engloba los dos conceptos que se vieron anteriormente, por lo que primeramente tenemos que escribir el polinomio, encontrar sus raíces con **roots**, graficarlo con **polyval** y finalmente graficar las raíces encontradas. Para graficar las raíces es importante considerar que, los valores que se grafican en ***x*** corresponden con los resultados de la función roots y los que se grafican en ***y*** simplemente son 0 (si no sabes por qué regresa al concepto de raíces de la página 4). El código quedaría de la siguiente forma:

```
P = [1, 0, -2, 1];
r = roots(P);
```

```
x= -2 : 0.01: 1.5;
y = polyval(P, x);
plot(x,y)
hold on
plot(r,0,'*r')
```



Aquí vemos que las raíces se marcan con asteríscos rojos y que coinciden con el cruce del polinomio por el eje de las x, éstas son:

```
-1.6180
 1.0000
 0.6180
```

- **polyfit (ajuste de datos)**

Al realizar algún experimento es común que los datos obtenidos los grafiquemos para evaluar su comportamiento, además, si queremos comprobar alguna Ley física que sabemos que explica el fenómeno del experimento debemos de encontrar una ecuación que ajuste a estos datos, es decir, una expresión matemática de la forma que se desee pero que pase por todos los datos con la mínima desviación.

En este caso podemos suponer que los datos podemos ajustarlo a una expresión polinomial de la forma:

$$P(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots + a_nx^n$$

Entonces, la desviación de esta con cada dato en ***x y*** obtenido del experimento sería:

$$Desv = y_i - a_0 - a_1x_i - a_2x_i^2 - a_3x_i^3 - \cdots - a_nx_i^n$$

Sin embargo, queremos que esta desviación sea mínima para todos los puntos x y del experimento por lo que hay que realizar la suma de toda la expresión **Desv**:

$$\sum Desv = \sum(y_i - a_0 - a_1x_i - a_2x_i^2 - a_3x_i^3 - \cdots - a_nx_i^n)$$

Sin embargo, para asegurar que se encontrará el mínimo de la expresión **Desv** podríamos elevar ésta al cuadrado, quedando:

$$\sum Desv^2 = \sum(y_i - a_0 - a_1x_i - a_2x_i^2 - a_3x_i^3 - \cdots - a_nx_i^n)^2$$

Una vez teniendo esta expresión hay que encontrar el mínimo utilizando el método de la primera derivada, en este caso las variables que nos podrán ayudar a encontrar el mínimo y que no conocemos son todos los coeficientes $\mathbf{a_i}$ por lo que se tendría que hacer la derivada parcial de **Desv** respecto a cada una de las $\mathbf{a_i}$. Una vez realizado esto, obtendremos un sistema de ecuaciones que al resolverlo nos dará los valores de todos los coeficientes.

El método descrito anteriormente se le llama "Mínimos Cuadrados" y sirve para calcular los coeficientes del polinomio grado n que ajusta a un conjunto de datos **x y**. Cabe mencionar que las expresiones para encontrar dichos coeficientes pueden ser bastante complejas dependiendo del grado del polinomio, sin embargo, en Matlab podremos hacer el cálculo usando el comando polyfit que tiene la siguiente sintaxis:

P = polyfit(x,y,n)

Donde: ***P*** será el vector que contendrá el polinomio grado ***n*** de ajuste por mínimos cuadrados (considerando la notación de polinomios que hemos revisado) de los valores ***x*** y ***y***.

Ejemplo:

9.- Suponga que los datos mostrados en la siguiente tabla se obtuvieron en un experimento del laboratorio de física. Encontrar el polinomio grado 1 de ajuste de los datos por mínimos cuadrados. En una misma figura graficar los datos experimentales como asteriscos rojos y el polinomio de ajuste como una línea azul.

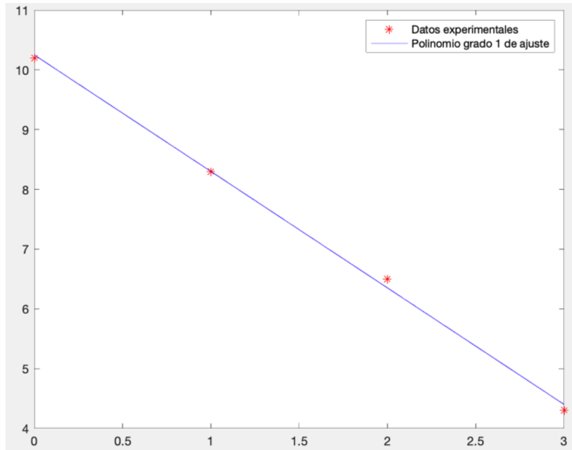
x	0	1	2	3
y	10.2	8.3	6.5	4.3

Solución

En este caso tenemos que introducir los datos de la tabla como vectores, posteriormente calcular el polinomio grado 1 de ajuste (una línea recta) utilizando **polyfit** con n=1 y graficar los datos y el polinomio encontrado usando el comando **polyval**. El código quedaría de la siguiente forma:

```
%Escribiendo los datos de la tabla
x = [0, 1, 2, 3];
y = [10.2, 8.3, 6.5, 4.3];

%Encontrando el polinomio de ajuste
P = polyfit(x,y,1)

%Graficando los datos de la tabla
plot(x,y,'*r')

%Graficando el polinomio de ajuste
X=0 : 0.01 : 3;   %Creamos un vector en el mismo intervalo que los datos
iniciales
Y=polyval(P,X);  %Evaluamos el polinomio P en los datos de X

hold on             %Graficamos
plot(X,Y,'-b')

%Etiquetando las gráficas
legend('Datos experimentales','Polinomio grado 1 de ajuste')
```

Así es como obtenemos la siguiente gráfica



Podemos observar que el polinomio de ajuste de grado 1 es una línea recta que pasa muy cercana a todos los puntos de la tabla.

El programa además despliega los valores de P que son los siguientes:

```
-1.9500   10.2500
```

Considerando entonces que estos siguen la misma notación de polinomios que hemos repasado en este material, sabemos que ese vector representa al siguiente polinomio:

$$P(x) = -1.9500x + 10.2500$$

Ejemplo:

10.- Suponga que los datos mostrados en la siguiente tabla se obtuvieron en un experimento. Encontrar los polinomios de grado 2 y 3 que ajusten a los datos por mínimos cuadrados. En una misma figura graficar los datos experimentales como asteriscos rojos y los polinomios de ajuste.

x	0	1	2	3
y	10.2	12.6	13.7	11.9

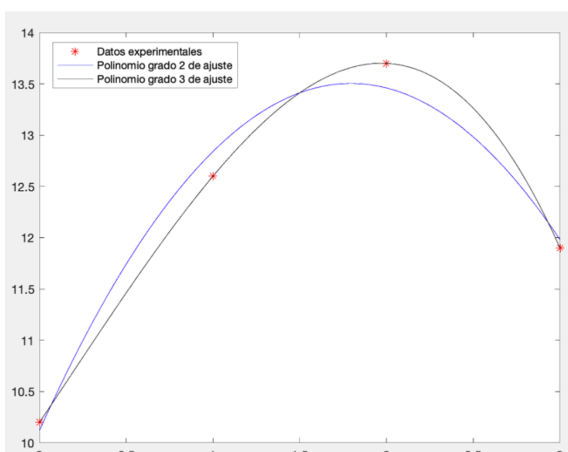
Solución

En este problema tenemos que realizar lo mismo que el anterior considerando que ahora tendremos que utilizar dos veces el comando **polyfit** una para n=2 y otra para n=3. Entonces el código quedaría de la siguiente forma:

```
%Escribiendo los datos de la tabla
x = [0, 1, 2, 3];
y = [10.2, 12.6, 13.7, 11.9];
```

```
%Encontrando el polinomio de ajuste grado 2
P2 = polyfit(x,y,2)

%Encontrando el polinomio de ajuste grado 3
P3 = polyfit(x,y,3)

%Graficando los datos de la tabla
plot(x,y,'*r')

%Graficando el polinomio de ajuste grado 2
X=0 : 0.01 : 3;   %Creamos un vector en el mismo intervalo que los datos
iniciales
Y=polyval(P2,X);  %Evaluamos el polinomio P en los datos de X

hold on           %Graficamos
plot(X,Y,'-b')

%Graficando el polinomio de ajuste grado 3
Y=polyval(P3,X);  %Evaluamos el polinomio P en los datos de X
plot(X,Y,'-k')

%Etiquetando las gráficas
legend('Datos experimentales','Polinomio grado 2 de ajuste','Polinomio grado 3
de ajuste')
```

Así es como obtenemos la siguiente gráfica



Podemos observar que el polinomio de ajuste de grado 2 forma una parábola (línea azul) que pasa cerca de los datos iniciales, sin embargo, el polinomio de grado 3 (línea negra) es el que mejor ajusta a los datos.

Los valores de P2 que despliega el programa son los siguientes:

```
-1.0500    3.7700   10.1200
```

Este vector representa al siguiente polinomio:

$$P_2(x) = -1.0500x^2 + 3.7700x + 10.1200$$

Por otra parte, los valores de P3 son:

```
-0.2667    0.1500    2.5167   10.2000
```

Por lo tanto:

$$P_3(x) = -0.2667x^3 + 0.1500x^2 + 2.5167x + 10.2000$$