Tutorial de Polinomios de Lagrange en MATLAB

Contents

1. MATLAB manipula polinomios almacenado

sus coeficientes en un vector renglon (matriz 3 x 1) Asi el polinomio x^2 + 2*x +3 se almacenaria como el vector [1 2 3]

```
P = [1 2 3]
```

```
P =

     1     2     3
```

2. MATLAB posee diversas rutinas que nos permiten

manipular polinomios. Podemos evaluar polinomios en puntos particulares con el comando 'polyval'. La siguiente instruccion evalua al polinomio con coeficientes almacenados en P arriba en x = 2:

```
polyval(P,2)
```

```
ans =

    11
```

3. El comando 'conv' se puede usar para multiplicar

dos polinomios. Por ejemplo, para multiplicar al polinomio de arriba por (x + 1); es decir, calcular (x^2 + 2x + 3)*(x + 1):

```
Q = [1 1]
conv(P,Q)

% (el resultado es entonces x^3+ 3*x^2 + 5*x + 3)
```

```
Q =

     1     1


ans =

     1     3     5     3
```

4. La caja de herramienta (Toolbox) Simbolica de MATLAB

tambien posee algunos comandos para maipular polinomios de forma simbolica. El comando 'poly2sym' convierte a un vector de coeficientes en un polinomio simbolico como sigue:

```
syms x
poly2sym(P,x)
```

```
ans =

x^2 + 2*x + 3
```

5. Para hallar al polinomio de grado 'n', que pasa

a traves de n + 1 puntos, podemos emplear 'polyfit':

```
help polyfit

% Por ejemplo, para hallar al polinomio que pasa por
% los puntos (X,Y):

X = [1 2 3 5]
Y = [1.06 1.12 1.34 1.78]
P = polyfit(X,Y,3)

% Asi, el polinomio que ajusta en estos puntos es
% -0.02*x^3 + 0.2*x^2 - 0.4*x + 1.28
%
% Se puede observar que MATLAB produce los coeficientes
% correctos de acuerdo con la teoria.
%
% La fucneion 'polyfit' emplea metodos del algebra lineal
% para resolver sistemas lineales de ecuaciones para hallar
% los coeficientes.
%
```

```
% Nosotros emplearemos el metodo de Lagrange que nos dara
% el mismo resultado, pero con mejor precision ante el redondeo
% computacional.
```

```
 POLYFIT Fit polynomial to data.
    P = POLYFIT(X,Y,N) finds the coefficients of a polynomial P(X) of
    degree N that fits the data Y best in a least-squares sense. P is a
    row vector of length N+1 containing the polynomial coefficients in
    descending powers, P(1)*X^N + P(2)*X^(N-1) +...+ P(N)*X + P(N+1).

    [P,S] = POLYFIT(X,Y,N) returns the polynomial coefficients P and a
    structure S for use with POLYVAL to obtain error estimates for
    predictions.  S contains fields for the triangular factor (R) from a QR
    decomposition of the Vandermonde matrix of X, the degrees of freedom
    (df), and the norm of the residuals (normr).  If the data Y are random,
    an estimate of the covariance matrix of P is (Rinv*Rinv')*normr^2/df,
    where Rinv is the inverse of R.

    [P,S,MU] = POLYFIT(X,Y,N) finds the coefficients of a polynomial in
    XHAT = (X-MU(1))/MU(2) where MU(1) = MEAN(X) and MU(2) = STD(X). This
    centering and scaling transformation improves the numerical properties
    of both the polynomial and the fitting algorithm.

    Warning messages result if N is >= length(X), if X has repeated, or
    nearly repeated, points, or if X might need centering and scaling.

    Class support for inputs X,Y:
       float: double, single

    See also POLY, POLYVAL, ROOTS, LSCOV.

    Reference page in Help browser
       doc polyfit
```

```
X =

     1     2     3     5


Y =

    1.0600    1.1200    1.3400    1.7800


P =

   -0.0200    0.2000   -0.4000    1.2800
```

Construyendo el Polinomio Interpolante de Lagrange

6. A fin de construir los coeficientes de Lagrange

del polinomio de Lagrange en MATLAB, podemos emplear la funcion 'poly' construida con este

proposito, la que construye un polinomio dadas sus raices.

Por ejemplo, tecleamos lo siguiente para construir un polinomio con raices x = 1 y x = 2:

```
poly([1 2])

% Asi el polinomio es el siguiente
% (x - 1)*(x - 2) = x^2 - 3*x + 2
% el cual posee raices en 1 y 2.

% Ejemplo.
% Polinomio interpolante de Lagrange de grado 2
% (cuadratico) que aproxima a la funcion
% f(x) = 1 + 2/x
% en los puntos x0 = 1, x1 = 2 y x3 = 2.5
%
% Observando en nuestra formula del coeficiente
% polinomial de Lagrange L2,0 podemos ver que
% necesitamos que el numerador sea el polinomio
% con raices x1 = 2 y x2 = 2.5
% (x - 2)*(x - 2.5)
% El denominador es la constante
% (x0 - x1)*(x0 - x2) = (1 - 2)*(1 - 2.5)
```

```
ans =

     1    -3     2
```

7. Supongamos que deseamos guardar los coeficientes

polinomiales de Lagrange en un arreglo 3x3, que llamaremos 'L' (con el 1er renglon siendo el coeficiente L2,0, el 2o renglon el coeficiente L2,1 y asi el 3o, L2,2), procedemos entonces como sigue:

```
L(1,:) = poly([2.0 2.5])/((1.0 - 2.0)*(1.0 - 2.5))

L(2,:) = poly([1.0 2.5])/((2.0 - 1.0)*(2.0 - 2.5))

L(3,:) = poly([1.0 2.0])/((2.5 - 1.0)*(2.5 - 2.0))
```

```
L =

    0.6667   -3.0000    3.3333
   -2.0000    7.0000   -5.0000
    1.3333   -4.0000    2.6667


L =

    0.6667   -3.0000    3.3333
   -2.0000    7.0000   -5.0000
    1.3333   -4.0000    2.6667
```

```
L =

    0.6667   -3.0000    3.3333
   -2.0000    7.0000   -5.0000
    1.3333   -4.0000    2.6667
```

8. El polinomio de Lagrange final es

y0*L2,0(x) + y1*L2,1(x) + y2*L2,2(x) Dado que y0 = f(x0) = 1.0 + 2.0/1.0 = 3.0 y similarmente y1 = 3.0, y2 = 3.3, calculamos al polinomio P como sigue

```
P = 3.0*L(1,:) + 3.0*L(2,:) + 3.3*L(3,:)
```

```
P =

    0.4000   -1.2000    3.8000
```

9. Para visualizar simbolicamente a este

polinomio en x tecleamos:

```
pretty(poly2sym(P))
```

```
     2
  2 x     6 x
  ---- - --- + 19/5
   5       5
```

10. Para evaluar a este polinomio en x = 1.5 usamos:

```
polyval(P,1.5)
```

```
ans =

    2.9000
```

11. Comparamos con el valor 'verdadero' de

f(x) = x + 2/x at 1.5

```
1.5 + 2.0/1.5
```

```
ans =

    2.8333
```

Similarmente para x = 1.2:

```
polyval(P,1.2)

1.2 + 2.0/1.5
```

```
ans =

    2.9360


ans =

    2.5333
```

12. Podemos graficar ambos resultados para compararlos.

Primero creamos al polinomio simbolico a partir de los coeficientes en P:

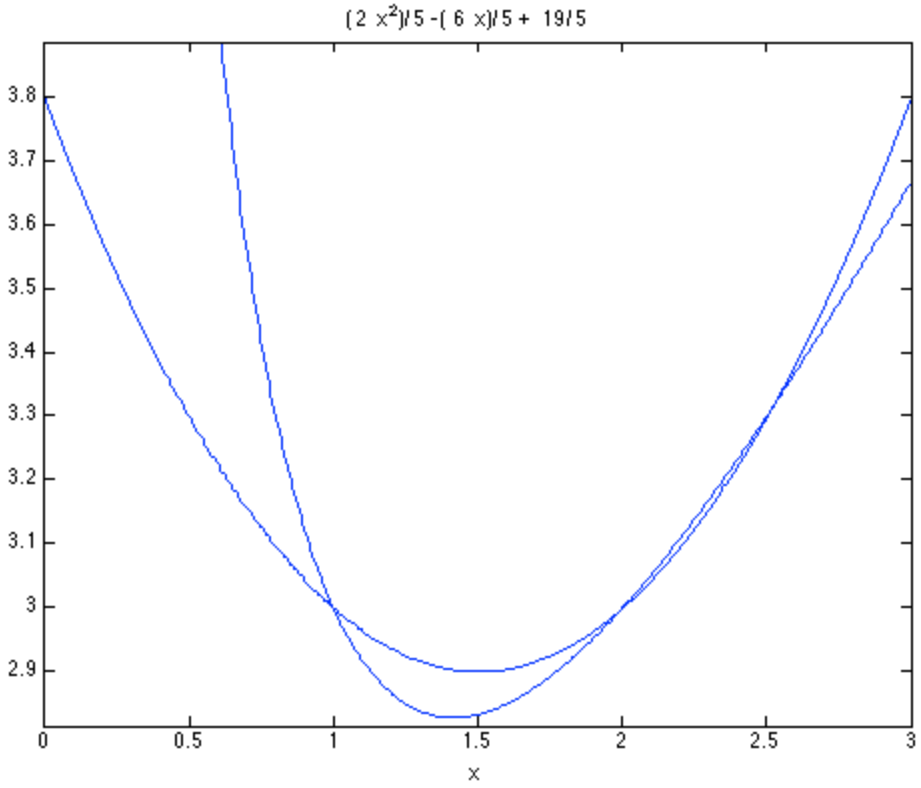
```
SP = poly2sym(P)
```

```
SP =

(2*x^2)/5 - (6*x)/5 + 19/5
```

13. Graficamos a la funcion f(x) = x + 2/x

en el intervalo de x = 0.5 a 3.0

```
figure
ezplot('x + 2/x', [0.5 3.0])

% Graficamos al polinomio interpolante de Lagrange SP
% sobre el mismo intervalos sobre la misma grafica:
hold on
ezplot(SP,[0.0 3.0])
```



El codigo para implementar un Algortimo para

calcular el polinomio interpolante de Lagrange.

14. Abrir el archivo 'lagran.m' en el editor de MATLAB

Note la interface para la funcion 'lagran':

```
% function [C,L]=lagran(X,Y)
%
% Input  - X is a vector that contains a list of abscissas
%        - Y is a vector that contains a list of ordinates
% Output - C is a matrix that contains the coefficents of
%          the Lagrange interpolatory polynomial
%        - L is a matrix that contains the Lagrange
%          coefficient polynomials
```

15. Asi, para llamar esta funcion primero establecemos

o asignamos a los vectores X y Y con los coeficientes 'x' y 'y' de los puntos de interpolacion. Entonces llamamos a la funcion que regresa al polinomio interpolante en la variable 'C', y a los coeficientes de Lagrange en la variable 'L'.

De nuestro ejemplo de arriba, tendriamos que

```
X = [1.0 2.0 2.5]
Y = [3.0 3.0 3.3]
[C L] = lagran(X,Y)
```

```
% Observamos que obtenemos la misma respuesta que antes.

% Veamos como es que son calculados los coeficientes
% en el cuerpo de la funcion:

% for k=1:n+1     % Calculate each of n+1 Lagrange coefficient
%    V=1;         % Accumulate computations in V temporarily
%    for j=1:n+1
%                 % Multiply by (x - X(j))/(X(k) - X(j))
%       if k~=j   % Be sure to skip the k'th one
%          V=conv(V,poly(X(j)))/(X(k)-X(j));
%       end
%
%    end
%    L(k,:)=V;    % Store Lagrange coefficient in kth row of L
% end

% El paso final es calcular el polinomio interpolante:
C = Y*L

% Esto emplea multiplicacion matricial para calcular
% C =  Y * L, que no es dificil verificar empleando
% las reglas de la multiplicacion matricial que nos da
% el polinomio correcto. Por ejemplo, la primer entrada
% en C sera
Y(1)*L(1,1) + Y(2)*L(2,1) + Y(3)*L(3,1)
% que es el coeficiente correcto del termino x^2.
```

```
X =

    1.0000    2.0000    2.5000


Y =

    3.0000    3.0000    3.3000


C =

    0.4000   -1.2000    3.8000


L =

    0.6667   -3.0000    3.3333
   -2.0000    7.0000   -5.0000
    1.3333   -4.0000    2.6667


C =

    0.4000   -1.2000    3.8000
```

```
ans =

    0.4000
```

Published with MATLAB® 7.12