Alumnos:

Método de Newton para aproximar raíces de ecuaciones polinómicas

Supongamos que queremos realiza un paso de Newton con el polinomio

$$p(x) = x^4 - 5x^3 + 4x^2 - 3x + 2,$$

partiendo de la estimación inicial x_0 = 2. Para ello, necesitamos evaluar el polinomio y su derivada en x_0 = 2.

Escribo su expresión anidada

$$p(x) = ((((x-5)x+4)x-3)x+2)$$

y sustituyo en ella el valor de x, utilizando la siguiente disposición de los cálculos, conocida como regla de Ruffini:

$$\begin{array}{r|rrrrr} & 1 & -5 & 4 & -3 & 2 \\ 2 & & 2 & -6 & -4 & -14 \\ \hline & 1 & -3 & -2 & -7 & \vert -12 \end{array}$$

El último elemento del resultado, -12, es el valor del polinomio en 2 y los otros elementos son los coeficientes del cociente

$$q(x) = x^3 - 3x^2 - 2x - 7\ .$$

Calculamos la derivada $p'(x_o)=q(x_o)$ evaluando $q(x)$ en x_0 de nuevo por Ruffini:

$$\begin{array}{r|rrrr} & 1 & -3 & -2 & -7 \\ 2 & & 2 & -2 & -8 \\ \hline & 1 & -1 & -4 \, \vert & -15 \end{array}$$

Así, $p'(x_o)$ es -15 y el siguiente iterado de Newton, $x_1 = 2 - \frac{12}{15} = \frac{3}{5}$.

En MATLAB hemos creado un archivo `horner.m` que evalúa un polinomio y su derivada en cierto valor a admitiendo como variables de entrada un vector con los coeficientes del polinomio, ordenados de mayor a menor grado y el valor de a.

1.- Usando como referencia el programa `newton.m` y como auxiliar `horner.m` elaborar un programa `newton_pol.m` de MATLAB para hallar raíces de un polinomio aplicando Newton.

a) Copia a continuación las líneas del programa `newton_pol.m`

b) Comprueba que en el ejemplo se obtiene la raíz x=0.8023 y añade a continuación el resultado de cada iteración.

c) Utiliza el programa `newton_pol.m` para resolver el siguiente problema.

Se suministran 2700 kJ/kmol de calor a presión constante a cierta cantidad de vapor de agua con una temperatura inicial *Ti* = 400 K. Calcular la temperatura final T_f del sistema teniendo en cuenta que la ecuación que rige el proceso es

$$Q = \int_{T_i}^{T_f} C_p dT$$

siendo $C_p = 32{,}24 + 0{,}001924 \cdot T + 0{,}00001055 \cdot T^2 - 3{,}596 \cdot 10^{-9} \cdot T^3$ (en kJ/kmol.K) el calor específico y Q el calor suministrado.
Explica a continuación como resolver el problema usando tu fichero `newton_pol.m` y escribe el resultado de cada iteración.

2.- Sabemos por teoría que un polinomio de grado n tiene exactamente n raíces, contando la multiplicidad y la posibilidad de que sean complejas. ¿Cómo hallarlas todas?

El caso raíces complejas no supone trabajo adicional. Gracias a que MATLAB trabaja con complejos y el método de Newton también es valido sobre los números complejos, basta con iterar a partir de una estimación inicial compleja.

El caso de raíces múltiples lo resolvemos aplicando el Teorema del Factor, consecuencia inmediata del resto que afirma que a es raíz de $p(x)$ si x-a es factor de $p(x)$. Una vez determinada una raíz a de $p(x)$, separamos el factor (x-a) y nos quedamos con el cociente exacto $q(x)$. Las raíces de este polinomio también lo son de $p(x)$. En particular, si a es también raíz de $q(x)$, será raíz doble de $p(x)$, y así sucesivamente.

Podemos utilizar este procedimiento para aproximar todas las raíces de un polinomio. Dado un polinomio *p(x)* encontramos una aproximación a de una de sus raíces, por el teorema del resto tenemos $p(x) = q(x)(x-a) + p(a) \approx q(x)(x-a)$. De este modo, para aproximar nuevas raíces de *p(x)* podemos buscar ahora aproximaciones de las raíces de *q(x)*.

Este procedimiento se denomina *deflación*. Cada vez que determinamos una raíz, reducimos el grado del polinomio en una unidad separando el factor correspondiente, hasta llegar a un polinomio de grado 0 que no tiene más raíces.

a) Elabora un programa `cociente.m` de MATLAB, modificando convenientemente la salida de `honer.m`, que nos proporcione el cociente de la división de un polinomio por (x-a). Escribe el programa a continuación.

b) Elabora un programa `deflaccion.m` de MATLAB para hallar todas las raíces de un polinomio usando la idea de deflación. Este programa llamará a `newton_pol.m`, pasándole la última raíz obtenida como estimación inicial y a `cociente.m` para obtener cociente de dividir por (x-‘*raiz*’).

c) Verifica tu algoritmo calculando todas las raíces del polinomio del ejemplo inicial.

Raíces: