Desarrollo Taller 1

1. Método de Horner

La evaluación usando la forma monomial del polinomio de grado-*n* requiere al menos *n*-sumas y $(n^2+n)/2$ multiplicaciones, si las potencias se calculan mediante la repetición de multiplicaciones.
El algoritmo de Horner sólo requiere *n*-sumas y *n*-multiplicaciones. (Minimizar el número de multiplicaciones es lo más deseable porque necesitan mucha carga computacional y son inestables comparadas con la suma).

En el algoritmo propuesto se comprueba que el número de operaciones sea 2n donde n corresponde al grado del polinomio. Evidentemente al derivar se disminuye en uno el valor del grado del polinomio por lo cual quitamos 2 operaciones por cada vez que derivamos y aplicamos el algoritmo.

2.

1) La escogencia de un método numérico lo más adecuado posible para el planteamiento del problema en términos matemáticos.
2) Primero que todo, se necesita comprender bien el problema con el que se trata. Se requiere conocimiento en el área respectiva de las matemáticas, en este caso, en cálculo, para realizar un planteamiento adecuado en términos matemáticos. Además, es valioso tener conocimientos en programación, para llevar trabajar el modelo computacionalmente.
3) La desventaja sería el tener que desperdiciar diferente recursos probando diferentes soluciones.
4) El error de truncamiento.
5) Principalmente radica en obtener soluciones más rápidamente, minimizando errores en el proceso.
6) Ya que dependiendo de cómo se halla hecho, puede haberse encontrado una solución aproximada no adecuada para el problema en cuestión.

3. Se obtiene un resultado con error relativo muy pequeño incluso para errores pequeños (0.1) y valores iniciales cercanos al número (10). En general, a mayor valor inicial se procesan más iteraciones. Al decrecer el error, la solución tiende a presentar un error relativo de cero(0). En cuanto a precisión, cuando se generan muchas iteraciones, los valores de x varían en considerable magnitud entre sí y sólo en las últimas iteraciones se obtienen valores cercanos a la raíz en cuestión. Cuando se generan pocas iteraciones, los valores de x son muy cercanos entre sí. El algoritmo converge rápidamente para un valor inicial pequeño y un error mínimo, donde el criterio es que los valores x y y se encuentren a una distancia menor al error.

4. Se presenta en el respectivo archivo.

5. Propagación Error

Los errores numéricos se generan con el uso de aproximaciones para representar las operaciones y cantidades matemáticas.

Error absoluto: Es la diferencia entre el valor de la medida y el valor tomado como exacto. Puede ser positivo o negativo, según si la medida es superior al valor real o inferior (la resta sale positiva o negativa). Tiene unidades, las mismas que las de la medida.

EA = | P* - P |

Error relativo: Es el cociente (la división) entre el error absoluto y el valor exacto. Si se multiplica por 100 se obtiene el tanto por ciento (%) de error. Al igual que el error absoluto puede ser positivo o negativo (según lo sea el error absoluto) porque puede ser por exceso o por defecto. no tiene unidades.

ER = | P* - P| / P , si P =/ 0

Para el ejercicio propuesto se tienen algunas condiciones iniciales propuestas determinadas por unos errores en la medición, con valores de 0.1, por lo cual es imprescindible tener en cuenta este número a la hora de dar un resultado aproximado. Los errores relativos muestran un rango de error en el cual un dato recogido pudo o no ser correcto, en este caso vemos que en el punto medio T(3) el error es 0. A medida que avanza o disminuye el T en este punto, el error aumenta progresivamente, aunque esto se debe a que tenemos en cuenta la distancia recorrida desde un P inicial y no un Px-1.

6. Convergencia o número de operaciones de un algoritmo para algoritmos directos

En este caso el algoritmo propuesto desarrolla un método directo debido a que cumple la condición de desarrollar un número finito de operaciones. De acuerdo a esto podemos descartar el orden de convergencia y buscar este número de operaciones, ya que es el que nos da un valor aproximado sobre el cual podemos asumir que tan bueno o no es nuestro algoritmo.

Según se desarrolla el algoritmo, en cada iteración dividimos el número a la mitad, comportamiento clásico en las soluciones de tipo logarítmicas, hecho que se comprobó mediante el uso de herramientas gráficas en las cuales se evidencia dicho comportamiento. Igualmente una regresión desarrollada sobre los datos puede evidenciar que la gráfica que se ajusta más es la logarítmica, aunque hay que tener en cuenta como se ve en las gráficas que el hecho de hacer “techo” o “roof” sobre el logaritmo escalona los valores obtenidos.

13. El algoritmo está desarrollado sobre la base del método de Newton por lo cual presenta algunas diferencias sobre otros metodos iterativos. El número de operaciones realizadas para este método depende del modo con el que se calcule la potencia. En el algoritmo se proponen dos soluciones, el uso de la funcion pow() de la librería math.h o la función fastPrecisePow(). Esta última función permite calcular potencias muy grandes sin embargo puede presentar un error de truncamiento entre el desarrollo de los cálculos por lo cual dificulta el cálculo sobre una gran cantidad de iteraciones. Por lo cual se utiliza la función pow () con el fin de no restringir el intervalo aunque se hace la claridad en cuanto a la implementación.

Fórmula de aproximación

1. Se establece un X inicial arbitrario no mayor a 10.
2. Se establece la tolerancia con la cual se va a trabajar en este caso 1e-3
3. Se establece un X Mayor correspondiente a un valor mayor al esperado, en este caso es el mayor entero permitido por el lenguaje.
4. Si este X Mayor es mayor a la tolerancia se procede a calcular un nuevo rango en el que se encuentre la solución.
    a. Se establece un punto en en el cual se encuentre la respuesta, esta entre el valor inicial y el valor A (valor al que se le sacará la raíz) elevado a una potencia X inicial.
    b. Se establece un nuevo xMayor que va desde el punto inicial hasta el punto calculado previamente. Se toma como valores absolutos por si se desfasan entre ellos.
    c. Se actualiza el X inicial y se repite el proceso.
5. Cuando el xMayor sea muy pequeño < 1e-3 se tendrá una aproximación de la solución con 3 cifras significativas.

El intervalo de convergencia está dado por |X - Xi| < 1e-3 donde:

- X representa el valor del resultado esperado.
- Xi el valor calculado en cada iteración.