ANyP, Práctica 3.

Resolución de ecuaciones no lineales.

En general, las raíces de una ecuación no lineal $f(x) = 0$ no pueden ser obtenidas por fórmulas explícitas cerradas, con lo que no es posible obtenerlas en forma exacta. De este modo, para resolver la ecuación nos vemos obligados a obtener soluciones aproximadas a través de algún método numérico.

✍ Estos métodos son *iterativos*, esto es, a partir de una o más aproximaciones iniciales para la raíz, generan una sucesión de aproximaciones $x_0, x_1, x_2, \ldots$ que esperamos converjan al valor de la raíz α buscada.

✍ La velocidad de convergencia de un método es medida como sigue. Sea $\epsilon_n = x_n - \alpha$, para cada $n \geq 0$. Si existe un número p y una constante no nula C tales que

$$\lim_{n\to\infty} \frac{|\epsilon_{n+1}|}{|\epsilon_n|^p} = C,$$

se dice que la sucesión $\{x_n\}$ converge a α con *orden* p y con una *constante asintótica de error* C. En particular, para $p = 1, 2, 3$ la convergencia se dice *lineal, cuadrática* y *cúbica*, respectivamente.

✍ El proceso iterativo se continúa hasta que la aproximación se encuentra próxima a la raíz dentro de una tolerancia $\epsilon > 0$ preestablecida. Como la raíz no es conocida, dicha proximidad, medida por el error absoluto $|x_{n+1} - \alpha|$, no puede ser computada. Sin un conocimiento adicional de la función $f(x)$ o su raíz, el mejor criterio para detener las iteraciones consiste en proceder hasta que la desigualdad

$$|x_{n+1} - x_n| < \epsilon\,|x_{n+1}|$$

se satisfaga, dado que esta condición controla, en cierta medida, el error relativo en cada paso.

✍ Puede ocurrir en ciertas circunstancias que la desigualdad anterior nunca se satisfaga, ya sea por que la sucesión de aproximaciones diverge o bien que la tolerancia escogida no es razonable. En tal caso el método iterativo no se detiene nunca. Para evitar este problema se considera además un número máximo de iteraciones a realizarse. Si este número es excedido entonces el problema debe ser analizado con más cuidado.

✍ ¿Cómo se escoge un valor correcto para las aproximaciones iniciales requeridas por los métodos? No existe una respuesta general para esta cuestión. Para el método de bisección es suficiente conocer un intervalo que contenga la raíz, pero para el método de Newton, por ejemplo, la aproximación tiene que estar suficientemente cercana a la raíz para que funcione. En cualquier caso primeras aproximaciones iniciales para las raíces pueden ser obtenidas graficando la función $f(x)$. En este sentido la utilidad `gnuplot` resulta de enorme ayuda. Tambien se puede graficar usando la libreria `matplotlib` de `Python`.

En los siguientes ejercicios se implementarán los métodos numéricos como *funciones Python*. Con el fin de escribir funciones de propósito general, consideraremos que las mismas tienen entre sus argumentos a la función f involucrada, la cual pude ser entonces implementada por el usuario como una `funcion` con el nombre que quiera. Otros argumentos que requeriran estas funciones son los valores para las aproximaciones iniciales que necesite el método, una tolerancia para la aproximación final de la raíz y un número máximo de iteraciones. La raíz aproximada será el valor que devuelva la función.
Dado que el método puede fallar, conviene implementar una variable entera como clave de error para advertir al programa principal. Por convención tomaremos que si dicha clave es igual a cero entonces el método funcionó correctamente y el valor devuelto es la raíz aproximada dentro de la tolerancia prescrita. En cambio, si la clave de error es distinta de cero, entonces ocurrió un error. La naturaleza del error dependerá del método, pero un error posible, común a todos los métodos, es que el número máximo de iteraciones fue alcanzado.

🏴 **Otras condiciones de paro.** Otras condiciones de paro en las iteraciones, además de la expuesta en las notas, son:

$$|x_{n+1} - x_n| < \epsilon \quad \text{ó} \quad |f(x_n)| < \epsilon$$

Desafortunadamente, pueden surgir dificultades usando cualquiera de estos criterios de paro, que hacen que no sean útiles.

Método de bisección o dicotómico.

El método de bisección comienza con un intervalo $[a, b]$ que contiene a la raíz α donde $f(x)$ cambia de signo. Entonces se computa el punto medio

$x_0 = (a+b)/2$ del mismo y se determina en cual de los dos subintervalos $[a, x_0]$ o $[x_0, b]$ se encuentra la raíz analizando el cambio de signo de $f(x)$ en los extremos. El procedimiento se vuelve a repetir con el nuevo intervalo así determinado.

Ejercicio 1. Implementar el método de bisección como una función `Python`.

Ejercicio 2. Determinar las raíces de las siguientes ecuaciones con una cota para el error absoluto $< \frac{1}{2} \times 10^{-8}$.

$$a)\ \cos x - x = 0 \qquad b)\ x^3 + 4\,x^2 - 10 = 0$$

Indicar el número de iteraciones realizadas.

🏴 Recordar, como se vió en la teoría, que la sucesión $\{x_n\}_{n=0}^{\infty}$ generada por el procedimiento de bisección aproxima a la raíz α satisfaciendo la relación

$$|\epsilon_n| \leq \frac{b-a}{2^{n+1}}, \quad n = 0, 1, \ldots$$

y que, por lo tanto, el método está acotado por una sucesión que converge a cero linealmente con una constante asintótica $C = 1/2$.

Método de Newton-Raphson o de la tangente.

El método de Newton comienza con una aproximación inicial x_0 dada para la raíz α, a partir de la cual se genera la sucesión de aproximaciones x_1, x_2, ..., siendo x_{n+1} la abscisa del punto de intersección del eje x con la recta tangente a $f(x)$ que pasa por el punto $(x_n, f(x_n))$. Esto conduce a la fórmula de iteración

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 1, 2, \ldots$$

Ejercicio 3. Implementar el método de Newton como una función `Python`. Notar que no sólo se requiere como argumento la función f sino también su derivada f'.

Ejercicio 4. Determinar las raíces de las siguientes ecuaciones con una cota para el error relativo $< 5 \times 10^{-8}$.

$$a)\ \cos x - x = 0 \qquad b)\ x^3 + 4\,x^2 - 10 = 0$$

Indicar el número de iteraciones realizadas.

🏴 Como vimos en la teoría, se puede mostrar que, si $f'(\alpha) \neq 0$ (raíz simple), la convergencia del método se comporta como $|\epsilon_{n+1}| \propto |\epsilon_n|^2$ y, por lo tanto, la convergencia es cuadrática.

Método de la secante.

El método de la secante procede a partir de *dos* aproximaciones iniciales de la raíz α obteniendo la aproximación x_{n+1} como la abscisa del punto de intersección del eje x con la recta secante que pasa por los puntos $(x_{n-1}, f(x_{n-1}))$ y $(x_n, f(x_n))$. La fórmula de iteración es entonces

$$x_{n+1} = x_n - f(x_n)\frac{(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}, \quad n = 2, 3, \ldots$$

Ejercicio 5. Implementar el método de la secante como una función `Python`.

Ejercicio 6. Determinar las raíces de las siguientes ecuaciones con una cota para el error relativo $\epsilon < 5 \times 10^{-8}$.

$$a)\ \cos x - x = 0 \qquad b)\ x^3 + 4\,x^2 - 10 = 0$$

Indicar el número de iteraciones realizadas.

🏴 Si $f'(\alpha) \neq 0$, se puede mostrar que, para el método de la secante,

$$\lim_{n\to\infty} \left| \frac{\epsilon_{n+1}}{\epsilon_n\, \epsilon_{n-1}} \right| = \left| \frac{f''(\alpha)}{2\, f'(\alpha)} \right|.$$

y entonces, la convergencia del método se comporta como $|\epsilon_{n+1}| \propto |\epsilon_n|^{\frac{1+\sqrt{5}}{2}}$ y, por lo tanto, la convergencia es más rápida que la lineal pero no tanto como la cuadrática.

Ejercicio 7. Volver a calcular las raíces de las ecuaciones de los ejercicios 2, 4 y 6 usando las funciones `bisect`, `secant` y `newton` de la librería `scipy.optimize`.

Raíces múltiples.

En la demostración de la convergencia de los métodos de Newton y de la secante la hipótesis de que $f'(\alpha) \neq 0$ es fundamental. Esta hipótesis significa que la raíz buscada es una *raíz simple*. Con más generalidad, se dice que α es una *raíz de multiplicidad* m de la ecuación $f(x) = 0$, si $f(x)$ puede escribirse como

$$f(x) = (x-\alpha)^m\, g(x), \qquad \text{siendo } \lim_{x\to\alpha} g(x) \neq 0.$$

✍ Si f tiene m derivadas continuas, entonces α es una raíz de multiplicidad m de $f(x) = 0$ si y solo si

$$f(\alpha) = f'(\alpha) = \cdots = f^{(m-1)}(\alpha) = 0, \quad \text{pero } f^{(m)}(\alpha) \neq 0.$$

✍ Si bien $f(x)$ es tangente al eje x en toda raíz múltiple α, el gráfico de $f(x)$ cruzará al eje x en α sólo si la multiplicidad m de la raíz es impar.

🏴 Como se vió en la teoría, se puede mostrar que el método de Newton-Rapshon es linealmente convergente para raíces múltiples, con una constante asintótica de error $C = \frac{m-1}{m}$.

Ejercicio 8. Mostrar que la función $u(x) = f(x)/f'(x)$ tiene las mismas raíces que $f(x)$ pero todas simples. Esto permite recuperar la convergencia cuadrática del método de Newton en el caso de raíces múltiples.

Ejercicio 9. Verificar experimentalmente la velocidad de convergencia del método de Newton resolviendo la ecuación $x^4 - 4\,x^2 + 4 = 0$, la cual tiene una raíz doble en $\alpha = \sqrt{2}$. Compare el número de iteracciones requeridas, para la misma precisión y aproximación inicial, aplicando ahora el método a la función $u(x) = f(x)/f'(x)$.

Iteración de punto fijo.

El método de punto fijo requiere que se reescriba la ecuación $f(x) = 0$ en la forma

$$x = \phi(x)$$

y entonces, a partir de una aproximación inicial x_0, se obtiene la sucesión de aproximaciones $x_1, x_2, \ldots$ según

$$x_{n+1} = \phi(x_n), \qquad n = 1, 2, \ldots$$

✍ Del teorema de existencia y unicidad del punto fijo resulta la siguiente condición suficiente de convergencia del método: si $\phi(x)$ es continuamente diferenciable sobre algún intervalo abierto que contenga al punto fijo α y $|\phi'(\alpha)| < 1$, entonces el esquema de iteración converge a α si la aproximación inicial está "suficientemente cerca" del punto fijo.

Ejercicio 10. Implementar el método de punto fijo como una función `Python`. Tener presente que la función que es pasada por argumento es ahora $\phi(x)$ y no $f(x)$.

Ejercicio 11. La ecuación $x + \log x = 0$ tiene una raíz cuyo valor aproximado es $\alpha \simeq 0.5$. Pudiendo elegir entre las siguientes fórmulas iterativas:

$$\begin{aligned} &a)\ x_{n+1} = -\log x_n, \\ &b)\ x_{n+1} = e^{-x_n}, \\ &c)\ x_{n+1} = \frac{x_n + e^{-x_n}}{2}, \end{aligned}$$

indicar cuales de ellas *pueden* ser utilizadas y cual *debería* usarse. Determinar la raíz de la ecuación con tal fórmula con una tolerancia en el error relativo de 5×10^{-8}.

🏴 Si $\phi'(\alpha) \neq 0$, la convergencia del método se comporta como $|\epsilon_{n+1}| \propto |\epsilon_n|$ y, por lo tanto, la convergencia es lineal.

Método de Newton para ecuaciones algebraicas.

En el caso particular en que $f(x)$ es un polinomio, el método de Newton puede ser eficientemente implementado si la evaluación de $f(x_n)$ (y su derivada) es realizada por el método iterativo de Horner. En efecto, supongamos que $f(x)$ es un polinomio de grado m:

$$f(x) = a_0 + a_1\,x + a_2\,x^2 + \cdots + a_m\,x^m,$$

la evaluación de $f(x_n)$ por la regla de Horner procede computando

$$\begin{cases} b_m = a_m \\ b_k = a_k + b_{k+1} x_n \quad k = m-1, \ldots, 0 \end{cases}$$

siendo, entonces

$$b_0 = f(x_n),$$

en tanto que $f'(x_n)$ es computada haciendo

$$\begin{cases} c_m = b_m \\ c_k = b_k + c_{k+1} x_n \quad k = m-1, \ldots, 1 \end{cases}$$

siendo, entonces

$$c_1 = f'(x_n).$$

El método de Newton se reduce así a

$$x_{n+1} = x_n - \frac{b_0}{c_1}$$

El procedimiento resultante se conoce a menudo como método de Birge-Vieta.

Ejercicio 12. Implementar el método anterior como una función `Python`. Notar que los coeficientes del polinomio pueden ser dispuestos en un arreglo unidimensional de $(m+1)$ componentes, siendo m el grado del polinomio.

Ejercicio 13. Encontrar aproximaciones exactas de las raíces de las siguientes ecuaciones algebraicas dentro de una tolerancia para el error relativo de 5×10^{-8}.

$$a)\ x^3 - 2\,x^2 + \frac{1}{2} = 0, \qquad b)\ x^4 + x^3 + 3\,x^2 + 2\,x - 2 = 0.$$

Ejercicio 14. Cuando se desea computar la permitividad eléctrica efectiva ε_{ef} de una roca compuesta

de N minerales diferentes con fracciones volumétricas f_i, $i = 1, ..., N$ y permitividades eléctricas ε_i, $i = 1, ..., N$, un modelo utilizado es la *aproximación de potencial coherente*, que establece (asumiendo ciertas hipótesis que no detallamos)

$$\sum_{i=1}^{N} f_i \frac{\varepsilon_i - \varepsilon_{ef}}{\varepsilon_i + 2\,\varepsilon_{ef}} = 0$$

a) Escribir un programa que permita encontrar ε_{ef} para K componentes, utilizando cualquiera de los métodos ya implementados en la práctica, cuidando que se cumplan las hipótesis del mismo.

b) Resolver el caso en que $K = 3$ componentes, con $f_1 = 0.3$, $\varepsilon_1 = 2.5\,\varepsilon_0$, $f_2 = 0.45$, $\varepsilon_2 = 3.1\,\varepsilon_0$, $f_3 = 0.25$, $\varepsilon_3 = 4.2\,\varepsilon_0$. Recordar que $\varepsilon_0 \approx 8.85 \times 10^{-12}\,\mathrm{F \cdot m^{-1}}$.