Diseño Proyecto Final

Tema: Solución de ecuaciones no lineales

Profesora:

Eddy Herrera Daza

Integrantes:

Miguel Ángel Beltrán Rodríguez

Juan Sebastián Triana Pérez

Pontificia Universidad Javeriana
Facultad de ingeniería
Bogotá D.C.

2018

Descripción del proyecto

Con el fin de recopilar lo aprendido durante el semestre en la materia de análisis numérico se escogió el proyecto de crear un paquete en Rstudio. Este paquete se compone de librerías sobre los temas vistos (Solución a ecuaciones no lineales, interpolación, derivación, integración, ecuaciones diferenciales). Nuestro grupo se encargará de la creación de una librería con el tema de soluciones a ecuaciones no lineales más específicamente en la búsqueda de raíces de una función, se trabajará con una precisión de 8 decimales.

Para comenzar es importante tener algunos conceptos claros, como son que es una ecuación no lineal o que son las raíces, una ecuación no lineal es una ecuación que tienen dos o más incógnitas y las raíces son todos los elementos x perteneciente al dominio de una función tal que se cumpla: *f(x) = 0.* Los métodos que se trabajaran para encontrar estas raíces se clasifican en 2, métodos cerrados y métodos abiertos. Los métodos cerrados parten de un intervalo en el que se sabe que hay al menos una raíz y siempre convergen, su método será la bisección. Los métodos abiertos parten de una aproximación inicial y tienen un radio de convergencia, sus métodos son el método del punto fijo, método de newton, el método de la secante y el método de la falsa posición.

Funciones de la librería

I. Bisección

Descripción: Encuentra raíces de funciones univariadas en intervalos limitados.
Uso:

biseccion(a, b, tol, f, precision)

Argumentos:

f: función continua
a: punto “menor” del intervalo cerrado
b: punto “mayor” del intervalo cerrado
tol: tolerancia

Detalles:

Bisección es un algoritmo de búsqueda de raíces para funciones continuas, univariadas y reales.
Bisección funciona dependiendo de si f (a) y f (b) tengan signos opuestos (es decir, $f(a) \cdot f(b) < 0$), el valor cero sería un valor intermedio entre f (a) y f (b), por lo que con certeza existe un x_* en [a,b] que cumple $f(x_*)=0$. De esta forma, se asegura la existencia de al menos una solución de la ecuación f(x)=0.
La bisección se detiene cuando se alcanza la tolerancia deseada.

Salida:

Una aproximación *m* de un cero *x** de *f* en |a, b|.

Seudocódigo:

```
k = 0;
repetir
    m = a +0.5(b − a);
    si f (m) = 0 entonces
        retornar m; parar
    dx = (b − a)/2;
    si signo(f (a)) diferente signo(f (m)) entonces
        b = m;
    si no
        a = m;
     k = k +1;
mientras dx ≤ tol;
retornar m;
```

II. Punto fijo

Descripción: Encuentra la raíz de una función dado una aproximación inicial y un máximo de iteraciones.

Uso:

puntoFijo(f, x0, tol, maxItr)

Argumentos:

f: función continua
x0: aproximación inicial
tol: tolerancia
maxItr: máximo de iteraciones

Detalles:

El método del punto fijo busca un x tal que $x = g(x)$. Un número $x = x*$ que satisface esta ecuación se llama punto fijo de g. Se detiene cuando se alcanza la tolerancia deseada o se llega al máximo de iteraciones.

Salida:

Si hay convergencia, una aproximación *Xn*, de un punto fijo.

Seudocódigo:

```
k = 0;
repetir:
    x1 = f (x0);
    dx = |x1 − x0|;
    x0 = x1;
    k = k +1;
    mientras (dx ≤ tol) o (k > maxItr);
retornar x1
```

III. Newton

Descripción: Encuentra raíces de funciones univariadas.

Uso:

newton(f, tol, maxItr)

Argumentos:

f: función continua

tol: tolerancia

maxItr: máximo de iteraciones

Detalles:

Si f es una función tal que f, f' y f '' existen y son continuas en un intervalo I y si un cero x_* de f está en I, se puede construir una sucesión {xn} de aproximaciones, que converge a x_* bajo ciertas condiciones:

Si x0 está suficientemente cercano al cero x_*, entonces supongamos que h es la corrección que necesita x0 para alcanzar a x_*, es decir, x0 +h = x_* y f (x0 +h) = 0. Como 0 = f (x0 +h) ≈ f (x0) +h f '(x0) entonces 'despejamos' la corrección h, h ≈ − f (x0) / f '(x0) De esta manera, una aproximación corregida de x0 sería

x1 = x0 – (f (x0) / f '(x0)).

Se detiene cuando se alcanza la tolerancia deseada o se llega al máximo de iteraciones, además las derivadas son calculadas dentro de la función.

Salida:

Si la iteración converge, una aproximación *Xn* de un cero de *f* en [*a,b*] y una estimación del error. Se retorna en un arreglo, donde la posición 0 es la aproximación y la posición 1 es la estimación del error.

Seudocódigo:

```
k = 0;
xk = x0;
repetir
    dx = f(xk )/f'(xk ) ;
    xk+1 = xk −dx;
    xk = xk+1;
    k = k +1;
mientras dx ≤ tol o k ≤ maxItr;
retornar xk y dx
```

IV. Secante

Descripción: Encuentra raíces de funciones univariadas.

Uso:

secante(f, x0, x1, tol, maxItr)

Argumentos:

f: función continua
x0 y x1: Aproximaciones iniciales
tol: tolerancia
maxItr: máximo de iteraciones

Detalles:

Aunque el método de la secante es anterior al método de Newton, a veces se hace una derivación de este método basado en la iteración de Newton cambiando la derivada f ‘(xk) por una aproximación ,Iniciando con dos aproximaciones iniciales x0 y x1, en el paso k +1, xk+1 se calcula, usando xk y xk−1, como la intersección con el eje X de la recta (secante) que pasa por los puntos (xk−1, f (xk−1)) y (xk , f (xk))

Entonces, si f (xk)− f (xk−1) diferente de 0,

xk+1 = (xk−1 f (xk)− xk f (xk−1)) / (f (xk)− f (xk−1))

Sin embargo, para cuidarnos del fenómeno de cancelación (cuando xk ≈ xk−1 y f (xk)f (xk−1) > 0), rescribimos la fórmula como:

xk+1 = xk − f (xk)· ((xk − xk−1)) / (f (xk)− f (xk−1)), k ≥ 1,

Se detiene cuando se alcanza la tolerancia deseada o se llega al máximo de iteraciones.

Salida:

Si la iteración converge a un cero, una aproximación xk del cero.

Seudocódigo:

```
j = 0;
mientras      |xk − xk−1| > tol y j < maxItr hacer:
        x2 = x1 − f (x1)*( (x1 − x0) /f (x1)− f (x0) );
        x0 = x1;
        x1 = x0;
        j = j +1;
retornar x2;
```

V. Falsa posición

Descripción: Encuentra raíces de funciones univariadas.

Uso:

falsaPosicion(f, a, b, maxItr)

Argumentos:

f: función continua
a, b: puntos del intervalo
maxItr: máximo de iteraciones

Detalles:

Se calcular la recta secante que une los puntos extremos (a1, f (a1)) y (b1, f (b1)). Luego se determina el punto m en que esta recta corta el eje x y este valor entra a jugar el papel que en el método de bisección jugaba el punto medio.

La recta secante tiene que une los puntos (a1, f (a1)) y (b1, f (b1)) tiene ecuación

$y = f(a1) + (f(b1) - f(a1)) / (a1 - b1) * (x - a1)$

Al resolver y = 0 se despeja el valor de x, obteniendo:

$m = a1 - (f(a1)(b1 - a1)) / (f(b1) - f(a1)) \Rightarrow m = (a1\, f(b1) - b1\, f(a1)) / (f(b1) - f(a1))$

En este método no se conoce a priori el número de iteraciones requeridas para alcanzar la precisión deseada así que se usa un número máximo de iteraciones.

Salida:

Una aproximación *Xn* de un cero de *f*

Seudocódigo:

```
k = 0;
an = a, bn = b;
repetir
    xn = (an f (bn)−bn f (an) ) / (f (bn)− f (an)) ;
    e1 = xn − an;
    e2 = bn − xn;
    si f (xn)f (an) > 0 entonces
        an = xn;
    si no
        bn = xn;
    k = k +1;
    mientras (máx(e1,e2) ≤ |xk − xk−1| o k ≥ maxItr) ;
retornar xn
```

Bibliografía

Mora, W. (Enero de 2016). *Cómo utilizar R en métodos numéricos disponible en: https://www.researchgate.net/publication/303889162_Como_utilizar_R_en_metodos_numericos .*