Revisión 1

Ronald Cardona
Anderson Grajales
Sebastian Valencia
Julian Sanchez

15 de septiembre de 2018

1. Solución numérica de ecuaciones de una variable

En esa seccion se describen algunos de los algoritmos numéricos principales para resolver ecuaciones de una variable. Estos algoritmos nos permiten ya sea encontrar intervalos en los que existe una raíz de la ecuación $f(x) = 0$ ó encontrar propiamente la raíz.

1.1. Búsquedas Incrementales

Leer x_0, *delta*, *niter*

$fx0 \leftarrow f(x_0)$

si $fx0 = 0$ **entonces**

  x_0 es raíz

en otro caso

  $x_1 \leftarrow x_0 + delta$

  $contador \leftarrow 1$

  $fx1 \leftarrow f(x_1)$

  mientras $fx0 * fx1 > 0$ ***y*** $contador < niter$ **hacer**

    $x_0 \leftarrow x_1$

    $fx0 \leftarrow fx1$

    $x_1 \leftarrow x_0 + delta$

    $fx1 \leftarrow f(x_1)$

    $contador \leftarrow contador + 1$

  fin

  si $fx1 = 0$ **entonces**

    x_1 es raíz

  si no, si $fx0 * fx1 < 0$ **entonces**

    Hay raíz entre x_0 y x_1

  en otro caso

    Fracasó en *niter* iteraciones

  fin

fin

Algoritmo 1: Algoritmo de Búsqueda Incremental

1.2. Método de la bisección

Leer x_i, x_s, $tolerancia$, $niter$
$fxi \leftarrow f(x_i)$
$fxs \leftarrow f(x_s)$
si $fxi = 0$ **entonces**
  x_i es raíz
si no, si $fxs = 0$ **entonces**
  x_s es raíz
si no, si $fxi * fxs < 0$ **entonces**
  $x_m \leftarrow \frac{x_i + x_s}{2}$
  $fxm = f(x_m)$
  $contador \leftarrow 1$
  $error \leftarrow tolerancia + 1$
  mientras $error > tolerancia$ ***y*** $fxm \neq 0$ ***y*** $contador < niter$ **hacer**
    si $fxi * fxm < 0$ **entonces**
      $x_s \leftarrow x_m$
      $fxs \leftarrow fxm$
    en otro caso
      $x_i \leftarrow x_m$
      $fxi \leftarrow fxm$
    fin
    $x_{aux} \leftarrow x_m$
    $x_m \leftarrow \frac{x_i + x_s}{2}$
    $fxm \leftarrow f(x_m)$
    $error \leftarrow |x_m - x_{aux}|$
    $contador \leftarrow contador + 1$
  fin
  si $fxm = 0$ **entonces**
    x_m es raíz
  si no, si $error < tolerancia$ **entonces**
    x_m es aproximación a una raíz con una tolerancia = $tolerancia$
  en otro caso
    Fracasó en $niter$ iteraciones
  fin
en otro caso
  El intervalo es inadecuado
fin

Algoritmo 2: Método de la Bisección

1.3. Método de la regla falsa

Leer x_i, x_s, $tolerancia$, $niter$
$fxi \leftarrow f(x_i)$
$fxs \leftarrow f(x_s)$
si $fxi = 0$ **entonces**
  x_i es raíz
si no, si $fxs = 0$ **entonces**
  x_s es raíz
si no, si $fxi * fxs < 0$ **entonces**
  $x_m \leftarrow x_i - \frac{f(x_i)*(x_s-x_i)}{f(x_s)-f(x_i)}$
  $fxm = f(x_m)$
  $contador \leftarrow 1$
  $error \leftarrow tolerancia + 1$
  mientras $error > tolerancia$ ***y*** $fxm \neq 0$ ***y*** $contador < niter$ **hacer**
    si $fxi * fxm < 0$ **entonces**
      $x_s \leftarrow x_m$
      $fxs \leftarrow fxm$
    en otro caso
      $x_i \leftarrow x_m$
      $fxi \leftarrow fxm$
    fin
    $x_{aux} \leftarrow x_m$
    $x_m \leftarrow x_i - \frac{f(x_i)*(x_s-x_i)}{f(x_s)-f(x_i)}$
    $fxm \leftarrow f(x_m)$
    $error \leftarrow |x_m - x_{aux}|$
    $contador \leftarrow contador + 1$
  fin
  si $fxm = 0$ **entonces**
    x_m es raíz
  si no, si $error < tolerancia$ **entonces**
    x_m es aproximación a una raíz con una tolerancia $=$ $tolerancia$
  en otro caso
    Fracasó en $niter$ iteraciones
  fin
en otro caso
  El intervalo es inadecuado
fin

Algoritmo 3: Método de la Regla Falsa

1.4. Método de Punto Fijo

Leer $tolerancia$, x_a, $niter$
$fx \leftarrow f(x_a)$
$contador \leftarrow 0$
$error \leftarrow tolerancia + 1$
mientras $fx \neq 0$ ***y*** $error > tolerancia$ ***y*** $contador < niter$ **hacer**
    $x_n \leftarrow g(x_a)$
    $fx \leftarrow f(x_n)$
    $error \leftarrow |x_n - x_a|$
    $x_a \leftarrow x_n$
    $contador \leftarrow contador + 1$
fin
si $fx = 0$ **entonces**
    x_a es raíz
si no, si $error < tolerancia$ **entonces**
    x_a es aproximación con una tolerancia = $tolerancia$
en otro caso
    El método fracasó en $niter$ iteraciones
fin

Algoritmo 4: Método de Punto Fijo

1.5. Método de Newton

Leer tol, x_0, $niter$
$fx \leftarrow f(x_0)$
$dfx = f'(x_0)$
$contador \leftarrow 0$
$error \leftarrow tol + 1$
mientras $fx \neq 0$ ***y*** $dfx \neq 0$ ***y*** $error > tol$ ***y*** $contador < niter$ **hacer**
| $x_1 \leftarrow x_0 - \frac{fx}{dfx}$
| $fx \leftarrow f(x_1)$
| $dfx \leftarrow f'(x_1)$
| $error \leftarrow |x_1 - x_0|$
| $x_0 \leftarrow x_1$
| $contador \leftarrow contador + 1$
fin
si $fx = 0$ **entonces**
| x_a es raíz
si no, si $error < tol$ **entonces**
| x_a es aproximación con una tolerancia = $tolerancia$
si no, si $dfx = 0$ **entonces**
| x_1 es una posible raíz múltiple
en otro caso
| El método fracasó en $niter$ iteraciones
fin

Algoritmo 5: Método de Newton

1.6. Método de la Secante

Leer x_0, x_1, $niter$, tol
$fx_0 \leftarrow f(x_0)$
si fx_0 = *0* **entonces**
| *imprimir* x_0 es raíz
en otro caso
$fx_1 \leftarrow f(x_1)$
$contador = 0$
$error = tol + 1$
$denominador \leftarrow fx_1 - fx_0$
mientras $error > tol$ ***y*** $fx_1 \neq 0$ ***y*** $denominador \neq 0$ ***y*** $contador < niter$ **hacer**
$x_2 \leftarrow x_1 - \frac{fx_1*(x_1-x_0)}{denominador}$
$error \leftarrow \left|\frac{x_2-x_1}{x_2}\right|$
$x_0 \leftarrow x_1$
$fx_0 \leftarrow fx_1$
$x_1 \leftarrow x_2$
$fx_1 \leftarrow f(x_1)$
$denominador \leftarrow fx_1 - fx_0$
$contador \leftarrow contador + 1$
fin
si $fx_1 = 0$ **entonces**
| *imprimir* x_1 es raíz
si no, si $error < tol$ **entonces**
| *imprimir* x_1 es una aproximación con tolerancia = tol
si no, si $denominador = 0$ **entonces**
| *imprimir* probablemente existe una raiz multiple
en otro caso
| *imprimir* fracasó en $niter$ iteraciones
fin
fin

Algoritmo 6: Método de la Secante

1.7. Método de Raices Multiples

Leer x_0, *niter*, *tol*
$fx \leftarrow f(x_0)$
$dfx \leftarrow f'(x_0)$
$ddfx \leftarrow f''(x_0)$
$denominador \leftarrow dfx^2 - fx * ddfx$
$contador \leftarrow 0$
$error \leftarrow tol + 1$
mientras $error > tol$ ***y*** $fx \neq 0$ ***y*** $denominador \neq 0$ ***y*** $contador < niter$ **hacer**
| $x_1 \leftarrow x_0 - \frac{fx*dfx}{denominador}$
| $fx \leftarrow f(x_1)$
| $dfx \leftarrow f'(x_1)$
| $ddfx \leftarrow f''(x_1)$
| $denominador \leftarrow dfx^2 - fx * ddfx$
| $error \leftarrow |x_1 - x_0|$
| $x_0 \leftarrow x_1$
| $contador \leftarrow contador + 1$
fin
si $fx = 0$ **entonces**
| *imprimir* x_0 es raíz
si no, si $error < tol$ **entonces**
| *imprimir* x_0 es una aproximación con tolerancia = *tol*
en otro caso
| *imprimir* fracasó en *niter* iteraciones
fin

Algoritmo 7: Método de Raices Multiples

2. Metodos de optimizacion para la solución numérica de ecuaciones de una variable

En algunos casos es necesario optimizar algunos métodos numéricos con la finalidad de alcanzar una tolerancia en el menor numero de iteraciones posible. Es por esto, que acá se presenta un método de optimización a el método de Punto Fijo, que hace que la convergencia sea mucho mas rápida y exacta.

2.1. Método de Steffensen

```
// Tomado de:
   https://en.wikipedia.org/wiki/Steffensen%27s_method
```

Leer x_0, *niter*, *tol*
$error = 1$+tol
$contador = 0$
mientras $contador < niter$ ***y*** $error > tol$ **hacer**
   $x_1 = f(x_0)$
   $x_2 = f(x_1)$
   $f(x) = f(x_0)$
   $p = x_0 - \frac{(x_1 - x_2)^2}{x_2 - 2 * x_1 + x_0}$
   $error = |x - x_0|$
   $x_0 = p$
   $contador = contador + 1$
fin
si $error \leq tol$ **entonces**
   Hay una raíz en p
en otro caso
   El método fracasó después de *niter* iteraciones
fin

Algoritmo 8: Método de Steffensen

3. Método de eliminacion Gaussiana Simple