Métodos Iterativos para Equações Escalares

Prof. Americo Cunha

Universidade do Estado do Rio de Janeiro – UERJ

americo.cunha@uerj.br

www.americocunha.org

@AmericoCunhaJr

@AmericoCunhaJr @AmericoCunhaJr

$$f(x) = 0$$

Como obter uma solução para essa equação se não sabemos resolvê-la analiticamente?

$$f(x) = 0$$

Como obter uma solução para essa equação se não sabemos resolvê-la analiticamente?

Desista de obter um valor exato, construa uma aproximação numérica!

(com auxílio de um método iterativo)

A estrutura de um método iterativo

1. **(Inicialização)** Defina:
   - uma *regra de aproximação* para x^*
   - um *chute inicial* x_0

2. **(Iteração)** Construa uma *sequência de aproximações* para x^*:

$$x_0, x_1, x_2, \cdots, x_n, \cdots \quad .$$

Num método iterativo *"bem projetado"*, tem-se que

$$\lim_{n \to \infty} x_n = x^*$$

i.e., a *sequência de aproximações converge* para a raiz x^*.

Algoritmo genérico para um método iterativo

Input: f, x_0, `tol`, `maxiter`

$\texttt{iter} = 0$
$\texttt{Error} = \infty$
while termination criterion is not met (`Error`, `tol`, `maxiter`) **do**
    $\texttt{iter} = \texttt{iter} + 1$
    Construct x_n based on $x_{n-1}, \cdots, x_0$
    Estimate the `Error` based on x_n and x_{n-1}
end while

Output: x_n, `iter`

Experimento computacional: $e^{-x} - x = 0$

MainIterativeMethod.m

```
clc; clear;

g = @(x) exp(-x)*(x + 1)/(exp(-x) + 1);
x0 = -10.0; tol = 1.0e-9; maxiter = 10;
iter = 0; Error = Inf;

while Error > tol && iter < maxiter
    iter = iter + 1;
    xnew = g(x0);
    Error = abs(xnew-x0);
    x0 = xnew;
    printf([' iter = %3d   ',...
            ' root = %.16f ',...
            'Error = %.16f \n'],...
            iter,xnew,Error);
end
```

Características desejáveis

- *Eficiência:* necessita de um pequeno número de avaliações computacionais da função f;
- *Robustez:* raramente (ou nunca) falha. Se falhar, informa ao usuário (você);
- Requer uma *quantidade mínima de informações* sobre f (e.g. continuidade, derivadas etc);
- Requer que f satisfaça *propriedades mínimas de suavidade*;
- Facilmente *generalizável* para equações com várias variáveis.

Considerações práticas

- Em *teoria*, um método iterativo "bem projetado" encontra o *valor exato* de x^*, pois constrói uma sequência convergente, i.e., $x_n \to x^*$ quando $n \to \infty$;
- Na *prática*, um método iterativo "bem projetado" encontra uma *aproximação* para x^*, pois um computador não consegue iterar infinitas vezes, i.e., $x_n \approx x^*$ para $n \gg 1$;
- Como é necessário *interromper o processo iterativo* após um número finito de etapas, um *critério de parada* se faz necessário;
- Diversos critérios de parada podem ser utilizados para verificar a *(quase) convergência* de um método iterativo.

Alguns critérios de parada

O processo iterativo é *interrompido* após *n iterações*, se:

- *CP1 (máximo de iterações):*

$$n > \texttt{maxiter}$$

- *CP2 ("erro" absoluto):*

$$|x_n - x_{n-1}| < \texttt{tol}$$

- *CP3 ("erro" relativo):*

$$|x_n - x_{n-1}| < \texttt{tol}\,|x_n|$$

- *CP4 (teste do resíduo):*

$$|f(x_n)| < \texttt{tol}$$

Parâmetros:

- x_n e x_{n-1} são duas aproximações sucessivas para x^*;
- `tol` e `maxiter` são parâmetros definidos pelo usuário (você).

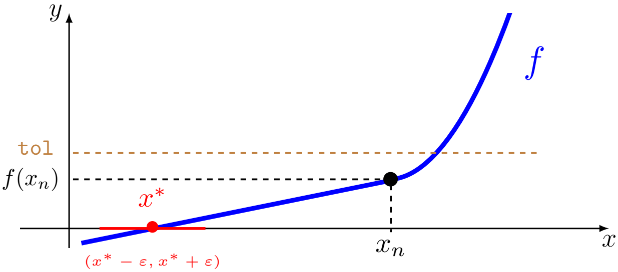
Observações sobre os critérios de parada

- Em geral (mas nem sempre) *(CP3)* é mais robusto que *(CP2)*;
- Os critérios *(CP2)* e *(CP3)* podem ser combinados:

$$|x_n - x_{n-1}| < \texttt{tol}\ \big(1 + |x_n|\big)$$

- O critério *(CP4)* não tem precisão garantida, pois

$$|f(x_n)| < \texttt{tol} \not\Longrightarrow |x^* - x_n| < \varepsilon$$



* Figura por Marcos Vinícius Issa.

Como citar esse material?

A. Cunha Jr, *Métodos Iterativos para Equações Escalares*, Universidade do Estado do Rio de Janeiro – UERJ, 2020.

@AmericoCunhaJr

@AmericoCunhaJr @AmericoCunhaJr