Utilizando Método de Newton-Raphson e Descida de Gradiente para encontrar as raízes e o mínimo, respectivamente, de determinadas funções

Mariana Bastos dos Santos

Agosto 2019

1 Introdução

Para a solução de problemas computacionais é necessário utilizar métodos numéricos. Esses métodos utilizam aproximação e diversas iterações para tentar convergir para a solução e por isso, tendem a ser mais genéricos e computacionalmente mais custosos. Sendo assim, esse trabalho propõe o uso do Método de Newton-Raphson e Descida de Gradiente encontrar as raizes de duas funções e a Descida de Gradiente para encontrar o mínimo das mesmas duas funções.

2 Conceitos Fundamentais

2.1 Método de Newton-Raphson

[1] O Método de Newton-Raphson é utilizado para encontrar aproximadamente as raizes de uma função de forma iterativa. Para o cálculo, utiliza a função $f(x)$ e sua derivada $f'(x)$, em pontos arbitrários x. A fórmula de Newton-Raphson (equação 1) consiste geometricamente em estender a linha tangente (calculada pela derivada $f'(x)$) no ponto atual x_i até que ela cruze zero, então definindo o próximo palpite x_{i+1}, para a abscissa daquele cruzamento zero.

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_{i)}} \tag{1}$$

2.2 Descida de Gradiente

O Método de Descida de gradiente é utilizado para encontrar o mínimo de uma função seguindo a direção do gradiente. Para isso itera a equaçáo 2. Essa equação pode ser utilizada para determinar, por exemplo, o menor custo de uma regressão linear.

$$x_{i+1} = x_i - \nabla f(x_i) \tag{2}$$

3 Implementação

Neste trabalho foram implementados os métodos de Newton-Rapshon e Descida de Gradiente. Para implementação foi utilizada a linguagem Python.

Para implementação do Método Newton-Raphson foi implementado o algoritmo 1 em que itera os valores da função $f(x)$ e de sua derivada S_i até encontrar o valor mais próximo da raiz. A variável β se refere a taxa de aprendizado ($0 < \beta < 1$). O critério de parada foi estabelecido para até 70 iterações do algoritmo, caso não encontre o valor aproximado da raiz.

Algorithm 1 Calcular o Método de Newton-Rapshon

$max \leftarrow 70$
$x_0 \leftarrow 0$
$f_0 \leftarrow f(x_0)$
while $f_i > 0$ **do**
  $S_i \leftarrow \frac{df}{dx}(x_i)$
  $x_{i+1} \leftarrow x_i + \beta * (-f_i) * \frac{1}{S_i}$
  $f_{i+1} \leftarrow f(x_{i+1})$
  if $i == max$ **then**
    $break$
  end if
end while

Para implementação da Descida de Gradiente foi implementado o algoritmo 2 em que itera os valores da derivada da função $f(x)$ representada pela

variável S_i. A variável β se refere a taxa de aprendizado ($0 < \beta < 1$). O critério de parada foi estabelecido para até 40 iterações do algoritmo, caso não encontre o valor mínimo da função.

Algorithm 2 Calcular Descida do Gradiente

$max \leftarrow 40$
$x_0 \leftarrow 0$
$f_0 \leftarrow f(x_0)$
while $f'_i > 0$ **do**
  $S_i \leftarrow \frac{df}{dx}(x_i)$
  $x_{i+1} \leftarrow x_i - \beta * S_i$
  $f'_{i+1} \leftarrow f'(x_{i+1})$
  if $i == max$ **then**
    $break$
  end if
end while

4 Experimentos e Resultados

Os algoritmos foram testados para as equações 3 e 4, mudando a taxa de aprendizado β de 0,1 até 1,0, tanto para o Método de Newton-Raphson quanto para a Descida de Gradiente.

$$f(x) = x^2, \text{ sendo } x_0 = 2 \tag{3}$$

$$f(x) = x^3 - 2x^2 + 2, \text{ sendo } x_0 = 2 \tag{4}$$

Tabela 1: Resultados do Método de Newton-Raphson para as equações 3 e 4

Método de Newton-Rapshon											
Função 3						**Função 4**					
i	***f(i)***	$\mathbf{x}_i$	β	$\mathbf{x}_0$	**max**	***i***	***f(i)***	$\mathbf{x}_i$	β	$\mathbf{x}_0$	**max**
64	0,01	0,0073	0,1	2	70	69	0,09	-0,823	0,1	2	70
30	0,01	0,08	0,2	2	70	69	0,01	-0,837	0,2	2	70
20	0,01	0,082	0,3	2	70	9	2	-0,039	0,3	2	70
14	0,01	0,088	0,4	2	70	69	1,02	0,98	0,4	2	70
12	0,01	0,073	0,5	2	70	52	0,82	1,346	0,5	2	70
9	0,01	0,081	0,6	2	70	2	0,81	1,337	0,6	2	70
7	0,01	0,093	0,7	2	70	40	1,74	-0,332	0,7	2	70
6	0,01	0,086	0,8	2	70	4	0,81	1,339	0,8	2	70
5	0,01	0,105	0,9	2	70	15	0,82	1,361	0,9	2	70

Tabela 2: Resultados da Descida de Gradiente para as equações 3 e 4

Descida de Gradiente											
Função 3						**Função 4**					
i	***f(i)***	x_i	β	x_0	**max**	***i***	***f(i)***	x_i	β	x_0	**max**
29	0,01	0,003	0,1	2	40	10	0,01	1,335	0,1	2	40
13	0,01	0,003	0,2	2	40	0	2	2	0,2	2	40
7	0,01	0,003	0,3	2	40	0	2	2	0,3	2	40
4	0,01	0,003	0,4	2	40	0	2	2	0,4	2	40
0	4	2	0,5	2	40	0	2	2	0,5	2	40
0	4	2	0,6	2	40	5	-inf	inf	0,6	2	40
0	4	2	0,7	2	40	5	-inf	inf	0,7	2	40
0	4	2	0,8	2	40	5	-inf	inf	0,8	2	40
0	4	2	0,9	2	40	5	-inf	inf	0,9	2	40

Observando a Tabela 4 é possível afirmar que a taxa de aprendizado 0,8 é a melhor para a Função 3, pois utiliza menor número de iterações i para alcançar o resultado. Em Relação a Função 4 aplicada ao mesmo método, observa-se que a taxa de aprendizado 0,6 é a mais adequada, pois com 2 iterações obtem valor aproximado da raiz da Função. É possível observar, ainda na Função 4 que as taxas 0,1, 0,2 e 0,4 não encontraram o valor da

raiz, o algoritmo executou o critério de parada (o algoritmo 1 para quando o número de iterações atinge o valor máximo de 70).

Para a Descida de Gradiente, observando a Tabela 2 é possível afirmar que para a Função 3 a taxa de aprendizado 0,4 se mostrou mais eficaz, pois comparando com as taxas 0,1, 0,2 e 0,3, que possuem mesmo valor de *f(i)* e x_i, é a que possui menor número de iterações (i). Em Relação a Função 4 aplicada ao mesmo método, observa-se que a partir da taxa de aprendizado 0,2 o algoritmo possui um valor de mínimo muito elevado, sendo igual a 2. Por este motivo, é mais preciso utilizar a taxa de aprendizado 0,1 com maior número de iterações, mas que possui um valor mínimo mais próximo a zero.

5 Conclusão

Esse trabalho propôs a implementação dos algoritmos do Método de Newton-Raphson e Descida de Gradiente, com o objetivo de encontrar, respectivamente, as raizes e mínimo das funções 3 e 4. Nos resultados observados na Seção 4 foi possível avaliar a influência da taxa de aprendizado no número de iterações executados por cada um dos Algoritmos 1 e 2 e no valor da função. Portanto, é preciso analisar a aproximação dos valores de *f(i)* a zero, pois deve-se ter o equilibrio tanto do número de iterações quanto do valor da função em relação a zero.

Referências

[1] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing.* Cambridge University Press, New York, NY, USA, 3 edition, 2007.