Universidade Federal de São Carlos

Campus Sorocaba  
27/04/2015

Newton 5D

**Francisco Guiraldelli      379840**

**Rafael Pereira                           380431**

Sumário

Introdução ..................................................................03

Funções utilizadas ..................................................................03

Execução ..................................................................03

Análise das funções ..................................................................03

Conclusão ..................................................................03

Apêndice ..................................................................03

**Introdução**

Neste projeto desenvolvemos dois algoritmos para encontrar raízes de funções em sistemas de grande porte ou esparsos.

Temos um programa principal que recebe as entradas do usuário, chama as funções que executam o método de Newton e o método da secante, recebem o resultado e exibe para o usuário final.

Para cada método temos uma função em arquivo separado que executa o método correspondente com controle de erro e iteração, e exibe a reta tangente ou secante da iteração atual em um gráfico.

A função que calcula o valor de xk+1 para o método de Newton é:



A função que calcula o valor de xk+1 para o método da secante é:



Após a criação dos algoritmos, executamos o programa com várias funções de teste como entrada, e analisamos os resultados retornados de acordo com alguns critérios como tempo de execução e número de iterações.

Comentamos os resultados obtidos e obtivemos algumas conclusões mostradas no decorrer do relatorio. Por fim, disponibilizamos os códigos fonte no apêndice.

**Funções utilizadas**

Foram utilizadas várias funções de teste com fontes distintas, utilizamos exercícios de lista, funções do livro Cálculo numérico – Aspectos teóricos e computacionais, e algumas encontradas em websites de matemática.

A seguir mostramos algumas funções que nos ajudaram na elaboração do relatório:











**Execução**

Para testar o programa utilizamos as funções mencionadas acima, alterando os dados de entrada, obtendo os seguintes resultados:



|  |  |  |
| --- | --- | --- |
| Erro | -2 | -5 |
| x0 Newton | -1 | 1 |
| x0 | -1 | 1 |
| x1 | 0 | 2 |
| Iterações Newton | 2 | 5 |
| Iterações Secante | 4 | 8 |
| Resultado Newton | -0.7682198 | -0.7682215 |
| Resultado Secante | -0.7682215 | -0.7682215 |
| Tempo Newton | 0.1826 | 0.32004 |
| Tempo Secante | 0.26449 | 0.47438 |



|  |  |  |  |
| --- | --- | --- | --- |
| Erro | -5 | -5 | -5 |
| x0 Newton | pi/4 | -5 | 5 |
| x0 | pi/4 | -5 | 5 |
| x1 | pi | 5 | 80 |
| Iterações Newton | 3 | 31 | 26 |
| Iterações Secante | 5 | 7 | 7 |
| Resultado Newton | 0.7390851 | 0.7390851 | 0.7390851 |
| Resultado Secante | 0.7390851 | 0.7390851 | 0.7390851 |
| Tempo Newton | 0.39384 | 1.6146 | 1.3677 |
| Tempo Secante | 0.38275 | 0.44576 | 0.44472 |



|  |  |  |
| --- | --- | --- |
| Erro | -4 | -4 |
| x0 Newton | 1.5 | 0.5 |
| x0 | 1 | -0.5 |
| x1 | 2 | 0.5 |
| Iterações Newton | 3 | 12 |
| Iterações Secante | 7 | - |
| Resultado Newton | 1.447414 | 0.00008007797 |
| Resultado Secante | 1.447414 | - |
| Tempo Newton | 0.29201 | 0.71833 |
| Tempo Secante | 0.44571 | - |



|  |  |  |
| --- | --- | --- |
| Erro | -6 | -6 |
| x0 Newton | 0 | 1.2 |
| x0 | 0 | 1.2 |
| x1 | 0.5 | 1.4 |
| Iterações Newton | 21 | 4 |
| Iterações Secante | 28 | 6 |
| Resultado Newton | 1.324718 | 1.324718 |
| Resultado Secante | 1.324718 | 1.324718 |
| Tempo Newton | 1.1184 | 0.34549 |
| Tempo Secante | 1.4854 | 0.42759 |



|  |  |
| --- | --- |
| Erro | -4 |
| x0 Newton | 0.1 |
| x0 | -5 |
| x1 | 3 |
| Iterações Newton | - |
| Iterações Secante | - |
| Resultado Newton | - |
| Resultado Secante | - |
| Tempo Newton | - |
| Tempo Secante | - |

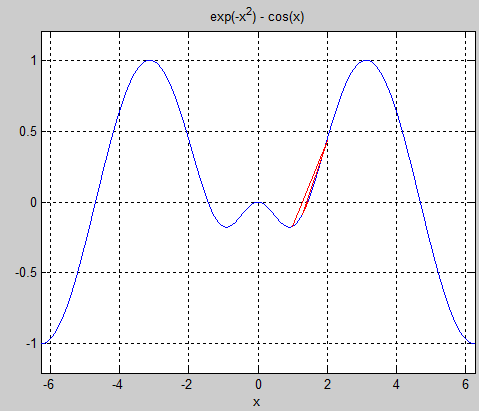
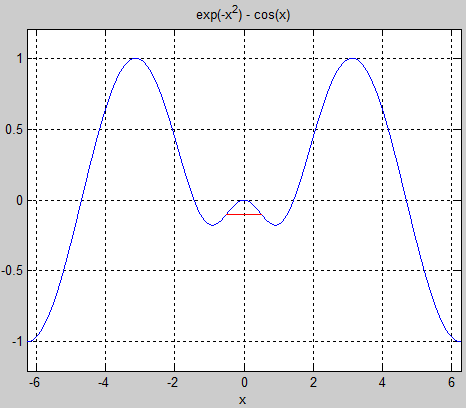
**Análise das Funções**

Após as execuções, analisando os dados obtidos nas tabelas anteriores e os gráficos, podemos estudar os valores obtidos e discorrer sobre a dependência de certas escolhas para um bom funcionamento dos algoritmos. Mostramos as análises a seguir:



Ao analisar esta função graficamente vemos que ela é uma função par [f(x) = f(-x)] e achamos interessante efetuar um segundo teste no método da secante com valores x0 e x1 com paridade distinta (x0 = -x1).

Partindo destes pontos, notamos que independentemente do número de iterações, a reta secante não é alterada, nunca chegando a um resultado final, como podemos mostrar nos gráficos a seguir:

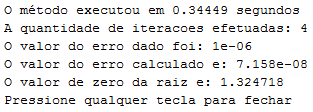
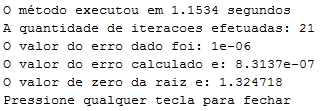
Método da secante com pontos distintos Método da secante com pontos opostos



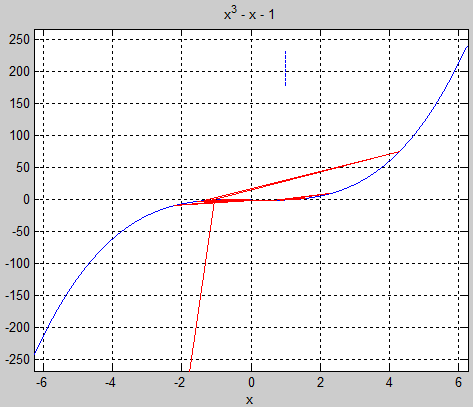
Esta função nos mostra a importância da boa execução da fase 1, onde achamos o ponto x0 e x1.

Com um ponto mais afastado da raiz da função, temos cinco vezes mais execuções dos métodos em média, do que com uma boa fase 1.

Podemos notar também que algumas retas secantes estão indo para fora do gráfico, aumentando demasiadamente o valor de um dos pontos, isto se dá pela aprximação de 0 de f(x-1)-f(x), a parte divisora do método, que ocasiona um número tendente ao infinito.



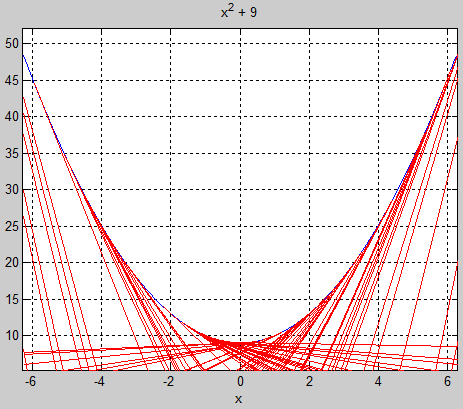
Fase 1 ruim para o método de Newton Fase 1 boa para o método de Newton



Explosão no cálculo da reta secante



Neste caso temos apenas uma função cuja f ”(x) existe, porém ela não possui raiz. Neste caso independentemente do número de iterações, a função nunca chegará a uma raiz aceitável também, sendo assim um desperdício computacional tentar executar o algoritmo com qualquer número de iterações.



Calculo do método de Newton em 100 iterações

**Conclusão**

Podemos notar que os métodos de Newton e da secante são robustos para encontrar raízes de funções em problemas grandes, esparsos ou com muitos números que tendem a dar 0 ao serem truncados.

Notamos rapidez na resolução dos problemas, visto que eram pequenos e não estávamos muito preocupados com o tempo de execução no momento, já que não estávamos aplicando o método em campo.

Notamos também que alguns casos particulares, como funções pares, exigem cuidado na escolha da fase 1, principalmente para o método da secante, e que uma boa fase inicial pode diminuir significativamente o número de execuções do algoritmo

**Apêndice**