Cálculo Diferencial e Integral I

Método de Newton

Resumo

Neste projeto iremos conhecer o método de Newton para encontrar zeros de funções e aplicá-lo para encontrar minimizadores de funções. Teremos um relatório em LaTeX, implementações em Julia e gráficos usando TikZ, a partir de algum programa.

Introdução

Dada uma função f continuamente diferenciável, buscamos um ponto x^* que seja um zero dessa função, ou seja, que $f(x^*) = 0$. Para polinômios de baixo grau, podemos usar algumas estratégias, mas existem equações não triviais, ou ainda que não tem solução fechada.

O algoritmo de Newton se resume a, dado uma aproximação para a solução x^k, encontrar a próxima aproximação fazendo

$$x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}$$

Como os pontos de mínimos são pontos críticos, e f é continuamente diferenciável, para encontrar um minimizador, buscamos um ponto tal que $f'(x) = 0$.

Questões

- Qual a importância de se encontrar zeros de função?
- Dê um exemplo de uma equação cuja solução não pode ser escrita com operações simples de uma maneira finita.
- Mostre como chegar na fórmula iterativa do método de Newton.
- Implemente o método de Newton na linguagem Julia.
- Teste seu método com alguns exemplos (mais de 10), incluindo algum que pode falhar dependendo do ponto inicial, e um que não tem solução.
- Mostre como encontrar $\sqrt{2}$ usando o método de Newton.
- Para a função $f(x) = x^2 - 1$, mostre (matematicamente) que a função converge para $x = 1$ se $x^0 > 0$ e para $x = -1$ se $x^0 < 0$.
- Discuta a sensibilidade ao ponto inicial para a função $f(x) = x^3 - x$ usando vários exemplos de ponto inicial.

Detalhes computacionais

Você deve usar o LaTeXpara escrever este projeto. Caso você não queira instalá-lo, você pode usar um dos sites:

- Sharelatex (`http://abelsiqueira.github.io/sharelatex-e-o-basico-do-cabecalho/`)
- Overleaf

- Authorea

Julia (`http://julialang.org/`) é bem fácil de instalar e usar, mas não deixe para última hora para aprender. Para fazer gráficos no Julia, você provavelmente irá precisar de um dos pacotes: PyPlot.jl, Gadfly.jl ou Plotly.jl. Provavelmente o PyPlot será suficiente. Sua entrega deve conter

- Um arquivo `newton.jl` contendo a implementação de uma função que busca a solução de $f(x) = 0$ usando o método de Newton. A função implementa deve receber a f, f' e x^0. Por exemplo, o código deve aceitar

  ```
  f(x) = x^2 - 1;
  fd(x) = 2*x;
  x, f, k, e = newton(f, fd, 2.3);
  ```

  onde `x, f, k, e` são, respectivamente, a aproximação para o zero da função, o valor da função nesse ponto, o número de iterações, e um número indicando o motivo da saída (0 = convergiu).

- Um arquivo `test.jl` contendo a implementação dos testes.

- Um arquivo `raiz2.jl` que calcula $\sqrt{2}$.

Uma maneira de fazer gráficos dentro do LaTeXé usando o TikZ, que é diretamente no código.

Para colocar códigos no LaTeX, é preciso usar um ambiente tipo o `verbatim`, ou o pacote `lstlisting`.

Entrega

A entrega deve ser um relatório em pdf e o .tex correspondente. Os códigos devem ser entregues no relatório e enviados por e-mail também. Os códigos serão testados e verificados, comparando o resultado ao escrito no relatório. Faça um relatório com introdução e conclusão, e as seções que forem necessárias para seu desenvolvimento.

Avaliação

O conteúdo será avaliado, assim como os resultados obtidos, a interpretação feita, e a validade do código. O código será rodado usando funções determinadas para o professor.

Dicas

- Um bom relatório não é apenas a resolução das perguntas, e sim um texto contínuo sobre o assunto.

- Indente o seu código.

- Verifique que seu código nunca vai ficar rodando pra sempre, e que pega os erros normais.

- Não deixe o código pra última hora. Na dúvida, procure o professor.

- Uma boa maneira de testar seu código é considerar uma família de funções geradas automaticamente cuja solução você conhece. Dessa maneira, você pode testar com centenas de funções ao invés de uma só. (Ainda conta como uma só para o objetivo acima).