

Uso da linguagem Python e do Google Colaboratory no ensino de cálculo numérico

Use of Python and Google Colaboratory in teaching numerical calculus

RESUMO

Enzo Dornelles Italiano
enzoitaliano@alunos.utfpr.edu.br
Universidade Tecnológica Federal
do Paraná, Campo Mourão, Paraná,
Brasil

Wellington José Corrêa
wcorrea@utfpr.edu.br
Universidade Tecnológica Federal
do Paraná, Campo Mourão, Paraná,
Brasil

Este manuscrito tem a finalidade de propiciar ferramentas auxiliares no estudo da disciplina de cálculo numérico. Para obter tal objetivo, foi definida a linguagem de programação *Python*, já abordada em outras disciplinas da UTFPR e que possui várias bibliotecas relacionadas à computação científica, assim como o aplicativo *app Google Colaboratory*, um host de *Jupyter Notebooks* que contém um interpretador de *Python*, assim como também consegue interpretar *markdown*, *Latex*, etc, tudo isso com recursos computacionais livres, podem assim confeccionar materiais interativos com código e texto para os alunos, tornando o processo de aprendizagem mais atrativo.

PALAVRAS-CHAVE: Python. Cálculo numérico. Ensino.

ABSTRACT

This manuscript has the purpose of providing auxiliary tools in the study of numerical calculus. To achieve this goal, the Python programming language was defined, already covered in other UTFPR disciplines and which has several libraries related to scientific computing, as well as the Google Colaboratory app, a host of Jupyter Notebooks that contains a Python interpreter, as well as being able to interpret markdown, Latex, etc., all with free computational resources, they can thus make interactive materials with code and text for students, making the learning process more attractive.

KEYWORDS: Python. Numerical calculus. Teaching.

Recebido: 19 ago. 2020.

Aprovado: 01 out. 2020.

Direito autoral: Este trabalho está licenciado sob os termos da Licença Creative Commons-Atribuição 4.0 Internacional.



INTRODUÇÃO

O Cálculo Numérico é uma ferramenta para solucionar problemas matemáticos recorrendo ao uso de um computador. Uma solução empregando Cálculo Numérico é numérica, ao passo que os procedimentos ditos analíticos normalmente apresentam um resultado proveniente de funções matemáticas. Além disso, uma solução numérica é uma aproximação do resultado exato. Diante dessa necessidade que a disciplina oferece, este manuscrito tem a finalidade de apresentar ferramentas de apoio ao ensino e aprendizagem de tópicos básicos de Cálculo Numérico, donde conceitos e resultados apresentados em sala de aula são explorados em aulas práticas nos laboratórios computacionais.

Especificamente, serão apresentados códigos na linguagem de programação *Python*, a serem usados na disciplina de Cálculo Numérico, incluindo métodos de busca de raízes de equações, solução de sistemas lineares, interpolações, integração numérica e solução numérica para equações diferenciais ordinárias. Sendo todos estes unidos e divididos em arquivos para leitura e execução pelo *Google Colaboratory*.

A escolha da linguagem *Python* foi imprescindível para o desenvolvimento do projeto, desde sua criação ela foi pensada como uma linguagem de sintaxe simples e tem seu foco no usuário avançado de computador mas, que não sejam programadores, fazendo com que os alunos não percam o interesse por ser uma linguagem de aprendizado mais rápido e fácil, visto que esta disciplina encontra-se em sua maioria, em cursos não relacionados diretamente à computação (PYTHON SOFTWARE FOUNDATION HISTORY, 2020).

A variedade de bibliotecas auxiliares para o *Python* também é um importante fator a ser mencionado, pois facilita na implementação dos métodos mais complexos do Cálculo Numérico, fazendo com que os arquivos fiquem menores e diminua o esforço do programador, sendo um dos objetivos da linguagem, o foco na solução do problema e não em questões técnicas, assim, é uma linguagem de alto nível.

O *Google Colaboratory* é um produto da *Google Research*, permitindo que qualquer pessoa com uma conta do *Google* acesse o aplicativo, escreva e execute qualquer código em *Python* diretamente do navegador sem a necessidade de instalar qualquer componente, a não ser alguma biblioteca que ele não contempla, não sendo muito comum, no entanto, a instalação ocorre apenas no próprio arquivo, tudo pelo navegador do computador ou pelo celular (GOOGLE COLABORATORY, 2020). Esta é a grande vantagem de nossa proposta em relação aos já consagrados softwares usados nos cursos de Cálculo Numérico como *VCN*, *Maple*, *Matlab*, *WxMaxima*, dentre outros no sentido em que não precisa instalar nenhum *software*, nem mesmo um período para o usuário conhecer e se adaptar com a interface e a sintaxe peculiar de cada *software* matemático.

MATERIAL E MÉTODOS

Os materiais necessários para o desenvolvimento do código foram o *Python* v3.8.3, assim como várias bibliotecas, algumas já instaladas junto ao *Python* como a *Copy* e a *Math*.

Bibliotecas externas são as que fornecem suporte à computação científica, foram instaladas utilizando a distribuição *Anaconda* v4.8.4 que visa simplificar o gerenciamento de pacotes *Python* no sistema, como *Matplotlib* v3.3.1, usada para geração de gráficos e nos dois últimos arquivos *Python* (relacionados aos tópicos finais da ementa de Cálculo Numérico) foi substituída pela *Plotly* v4.9.0; *SymPy* v1.6.1, utilizada para elementos de álgebra e também para substituição de vários métodos da biblioteca *Math* que não aceita elementos algébricos, em especial variáveis; *Numpy* v1.18.5, proporciona vários métodos para tratamento de dados e de computação científica. Além disso, foram utilizadas duas bibliotecas apenas para elementos visuais de matrizes e tabelas, sendo elas *PrettyTable* v0.7.2 para tabelas e *PrettyMatrix* v1.0.0 para matrizes.

A partir da fórmula de cada método a ser implementado, ofertado na ementa da disciplina, foram divididos em 6 listas para maior aproveitamento e aprendizado do aluno. É oportuno mencionar que as implementações criadas não fornecem apenas o resultado final. Por exemplo, existem softwares e bibliotecas que apresentam somente a função spline cúbica natural. Se um aluno errou a resposta apresentada, em um primeiro momento, ele não saberá onde cometeu o erro. Uma vez que estamos preocupados com o ensino e aprendizado, nossos códigos apresentam todos os passos de cada método (no caso da spline citada acima, o código apresenta todos os passos desde o sistema linear a ser resolvido, a solução do mesmo, os valores de todos os coeficientes dos polinômios da spline, bem como o seu gráfico) proporcionando ao aluno uma análise crítica, bem como descobrir aonde errou, caso sua resposta foi diferente da apresentada pelo *Google Colab*.

Depois de finalizada a implementação de cada método, ele passava por testes com argumentos e resultados já conhecidos para comprovação de funcionamento dele. Caso houvesse falha, o código da função era verificado linha por linha e testado após cada mudança, até que não apresentasse mais erros.

Após a verificação do código sem erros, foi montado um arquivo para o *Google Colaboratory* com todas as funções da lista finalizada.

RESULTADOS E DISCUSSÃO

Com a ideia de facilitar o estudo do aluno, a primeira proposta foi de criar função sobre cada método utilizado na disciplina de cálculo numérico. Ao final do desenvolvimento temos 6 arquivos com códigos em *Python* e arquivos do tipo *Jupyter Notebook* que podem ser abertos usando o *Google Colaboratory*.

Uma das funções implementadas foi a interpolação utilizando a fórmula de Lagrange colocado na lista de interpolação polinomial cuja fundamentação teórica pode ser encontrada em (FRANCO, 2006, p. 203) ou (BURDEN; FAIRES, 2003, p. 116).

Figura 4 – Função para o gráfico segundo o método de Lagrange

```

108 def graficoLagrange(pontos, valor, f):
109     Pn = 0
110     for i in range(len(pontos)):
111         mult = 1
112         div = 1
113         for j in range(len(pontos)):
114             if i == j: continue
115             mult *= P([-pontos[j][0], 1])
116             div *= pontos[i][0] - pontos[j][0]
117         Pn = Pn + pontos[i][1] * (mult // div)
118
119     fig, ax = plt.subplots()
120     z = np.arange(-0.5, 1.5, 0.01)
121
122     y = []
123     for i in range(len(z)):
124         y.append(Pn(z[i]))
125
126     a = []
127     for i in range(len(z)):
128         a.append(f.subs(x, z[i]))
129
130     b = []
131     w = []
132     for i in range(len(pontos)):
133         b.append(pontos[i][0])
134         w.append(pontos[i][1])
135
136     ax.plot(b, w, "r*", markersize=6, label="Pontos da tabela")
137     ax.plot(z, y, label='Polinômio Interpolador P(x)')
138     ax.plot(z, a, label="Função f(x)")
139     ax.plot(valor, Pn(valor), "g*", markersize=6, label="Estimativa")
140     ax.plot(valor, f.subs(x, valor), "yo", label="valor exato")
141     ax.legend()
142     ax.grid()
143     plt.show()

```

Fonte: Autoria própria (2020).

Com tal lista, estão incluídos alguns exemplos funcionais de exemplos de possíveis entradas:

Figura 5 – Exemplo de entrada

```

145 # def f(x): return (3+x)/(1+x)
146 # pontos = [[0.1, 2.82], [0.2, 2.67], [0.4, 2.43]]
147 # Lagrange(pontos, 0.25, f(x))
148 # graficoLagrange(pontos, 0.25)
149 # graficoLagrange(pontos, 0.25, f(x))

```

Fonte: Autoria própria (2020).

Utilizando uma simples execução pela lista do *Google Colaboratory* obteremos a saída representada na Figura 6 com o gráfico da Figura 7.

Figura 6 – Exemplo de saída

```
Polinômios coeficientes

>>>>>>>L[0]<<<<<<<
33.33333333333333*(x - 0.4)*(x - 0.2)

>>>>>>>L[1]<<<<<<<
-50.0*(x - 0.4)*(x - 0.1)

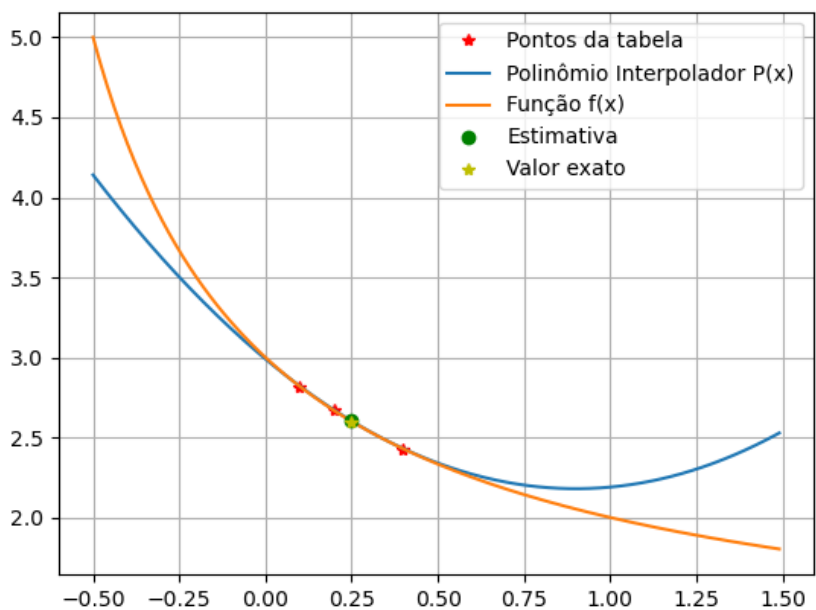
>>>>>>>L[2]<<<<<<<
16.66666666666667*(x - 0.2)*(x - 0.1)
Polinômio interpolador de Lagrange p(x) = 2.99 - 1.8x + 1.0x**2

Polinômio interpolador avaliado em x = 0.25 , é P(0.25) = 2.6024999999999974

Limitante
|E(0.25)| <= 0.00153678027457141
```

Fonte: Autoria própria (2020).

Figura 6 – Exemplo de saída



Fonte: Autoria própria (2020).

Vale enfatizar que tais ferramentas estão sendo usadas nas disciplinas de Cálculo Numérico dos cursos de Engenharia de Alimentos, Engenharia Química, Engenharia de Ambiental e Bacharelado em Ciências da Computação ministradas pelo professor Wellington José Corrêa da UTFPR, campus Campo Mourão seja na modalidade presencial (início do semestre) e na modalidade ADNP. O resultado tem sido excelente, donde os alunos apresentaram grande aceitação de tais ferramentas. Além da verificação de suas respostas nos exercícios propostos em

sala de aula e nas listas, os alunos recorrendo ao *Google Colaboratory* estão resolvendo problemas avançados relacionados a disciplina, explorando a capacidade e o bom êxito destas ferramentas.

CONCLUSÃO

Este trabalho teve como objetivo a implementação de métodos pertinentes a disciplina de Cálculo Numérico com foco a auxiliar o professor em despertar maior interesse nos alunos na parte computacional da disciplina e facilitar o estudo, seja em sala de aula ou em casa visto sua capacidade de apresentar não só o resultado mas como também suas iterações e etapas, sendo capaz de solucionar os problemas numéricos propostos na ementa do curso por conta da facilidade proporcionada pela linguagem. Assim, o projeto foi feito para uso livre, a fim de instigar e diminuir a dificuldade do aprendizado, tornando o processo mais interessante.

Os arquivos que são utilizados no *Google Colaboratory* podem ser encontrados no link <https://github.com/EnzoItaliano/calculoNumericoEmPython> onde se encontram todos os arquivos, tanto em *Python* como os do *Jupyter Notebook*.

AGRADECIMENTOS

Agradeço ao professor Wellington José Corrêa pela orientação e à Universidade Tecnológica Federal do Paraná pela estrutura.

REFERÊNCIAS

BURDEN, R. L.; FAIRES, J. D. **Análise Numérica**. São Paulo: Thompson Learning, 2003.

FRANCO, N. B. **Cálculo numérico**. São Paulo: Pearson Prentice Hall, 2006.

GOOGLE COLABORATORY. Disponível em: <https://colab.research.google.com/>. Acesso em: 04 set. 2020.

PYTHON SOFTWARE FOUNDATION HISTORY. Disponível em: <https://docs.python.org/3/license.html>. Acesso em: 04 set. 2020.