

DESENVOLVIMENTO DE E-BOOKS ESTÁTICOS E INTERATIVOS SOBRE GEOMETRIA ANALÍTICA E CÁLCULO VETORIAL PARA E- TUTORING EM CIÊNCIAS COMPUTACIONAIS E ENGENHARIA

Franklin Anthony Ramos Coêlho – franklinanthony@eng.ci.ufpb.br

Smith Lima da Silva – smithlima@ieee.org

Gustavo Charles Peixoto de Oliveira – gustavo.oliveira@ci.ufpb.br

Universidade Federal da Paraíba, Centro de Informática

Rua dos Escoteiros, S/N

58055-000 – João Pessoa – Paraíba

Resumo: O objetivo deste trabalho é apresentar e-books estáticos e interativos desenvolvidos pelo Programa de Educação Tutorial - PROTUT da Universidade Federal da Paraíba - UFPB para dar suporte ao ensino da componente curricular Cálculo Vetorial e Geometria Analítica em cursos de ciências computacionais e engenharia. Ferramentas utilizadas para a produção de texto, renderização de equações, plotagem de gráficos e escrita de códigos são discutidas com base no conceito de "caderno interativo" oferecido pelo Projeto Jupyter. O núcleo de programação interativa é executado integralmente em Python 3.x e usa módulos robustos de computação científica da linguagem, tais como Numpy, Scipy e Matplotlib.

Palavras-chave: E-book. Computação interativa. E-learning. Computação científica. Geometria Analítica.

1 INTRODUÇÃO

A presente década tem presenciado o enorme impacto que a tecnologia está causando nas formas de aprendizagem. A modalidade de aprendizado eletrônico e a distância (*e-learning*), quer *online*, na internet, quer *offline*, em conteúdo, elevou o papel do tutor a um patamar de suma importância, assim cunhando o conceito de *e-tutoring*. Diversos autores denominam "tutor" também como *facilitador*, *mediador*, ou mesmo como *coach*, além de descreverem as características desejáveis para o perfil. Um resumo dessas características pode ser encontrado em Klimova e Poulouva (2011).

Nos últimos anos, uma série de publicações veio à tona tendo em seu cerne não apenas o relato de experiências com *e-tutoring*, mas também a apresentação dos chamados *sistemas de tutoria inteligente* (STIs). Alguns deles, inclusive, foram aplicados às ciências exatas e, em particular, ao ensino de Matemática em diversos níveis de educação (CHEUNG et al., 2003; HOOSHYAR et al., 2018; KELES et al., 2009).

O *Programa de Educação Tutorial* (PROTUT) da Universidade Federal da Paraíba (UFPB) foi criado em 2014 e hoje faz parte do rol de políticas acadêmicas da universidade para o ensino em cursos de graduação. Distribuído pelos Centros de Ensino nos variados *campi*, o programa tem entre seus objetivos específicos os seguintes: (i) dar suporte didático-pedagógico para correção de desníveis de conhecimento prévio dos estudantes; (ii) auxiliar tutorandos no processo de aprendizagem; (iii) fomentar a confiança no estudo das disciplinas acadêmicas e (iv) colaborar para a melhoria de desempenho no processo de aprendizagem, visando à redução dos índices de reprovação, retenção e evasão.

Uma das ramificações do PROTUT está vinculada ao Centro de Informática (CI), onde se propõe a atender estudantes cuja formação adere-se à matemática e à computação. O PROTUT/CI age no suporte a seis disciplinas-alvo: três componentes curriculares versam sobre Cálculo Diferencial e Integral, uma sobre Cálculo Numérico, uma sobre Álgebra Linear e a uma sobre Cálculo Vetorial e Geometria Analítica (CVGA). Aqui daremos enfoque à última.

Este trabalho tem o propósito de apresentar e-books estáticos e interativos desenvolvidos pelo PROTUT da UFPB para dar suporte ao ensino de Cálculo Vetorial e Geometria Analítica em cursos de ciências computacionais e engenharia. Ferramentas utilizadas para a produção de texto, renderização de equações, plotagem de gráficos e escrita de códigos são discutidas com base no conceito de "caderno interativo" oferecido pela plataforma Jupyter.

2 METODOLOGIA

O *e-book* foi desenvolvido a partir de três processos (ver Figura 1), cada qual interdependente entre si: criação dos módulos, a partir da escrita do conteúdo propriamente dito, apresentando teoria e exemplos matemáticos; desenvolvimento dos algoritmos usando a linguagem de programação Python na resolução dos exemplos elencados em cada módulo e, por fim, a compilação, gerando o material final.

As ferramentas associadas à produção do material foram: *LaTeX*, que é uma linguagem compilada empregada na escrita de textos matemáticos e científicos de alta qualidade; *Markdown*, uma linguagem de marcação simples utilizada para a formatação de textos e tabelas, bem como estruturação de figuras. Juntamente com bibliotecas e APIs multi-browser, tais como o *Mathjax*, a tipografia matemática com Markdown torna-se bastante poderosa para a produção textual de alto desempenho; *Jupyter Notebook*, um ambiente integrado de escrita e desenvolvimento de algoritmos e *Google Colaboratory*, uma ferramenta *online* que permite a produção de material interativo utilizando Markdown e Python. Uma das principais

capacidades do Google Colab, como é conhecido, é a de permitir o desenvolvimento colaborativo do material, pelo qual diferentes alunos-tutores podem trabalhar de maneira simultânea, em tempo real.

Figura 1 - Fluxograma de desenvolvimento do *e-book*.



Fonte: Autores.

3 RESULTADOS E DISCUSSÃO

A Figura 2 mostra uma seção destacada da versão prévia do *e-book*. Nela pode-se observar que o processo de desenvolvimento do conteúdo foge à estrutura observada em livros didáticos no que se refere à teoria empregada em cada assunto. Como exemplo, é utilizada a abordagem da *distância entre dois pontos*, relacionada ao contexto de Cálculo Vetorial e Geometria Analítica. Após a dissertação de uma teoria concisa, é desenvolvido um exemplo sob uma ótica puramente matemática, o que dita uma relação entre teoria e prática, itens essenciais no processo ensino-aprendizagem. Logo após, é desenvolvida uma implementação baseada no mesmo exemplo exposto anteriormente, só que agora sob uma ótica algorítmica com o auxílio da linguagem Python, fazendo com que o aluno possa aplicar os conhecimentos adquiridos no uso de tal ferramenta computacional, acarretando em uma aprendizagem mais completa e multidirecional.

Ainda sobre a Figura 2, no que se refere à estrutura usada no desenvolvimento do *e-book*, pode-se observar: *título das seções* sobre o exemplo e a implementação envolvendo a distância entre dois pontos; *escrita textual* com o auxílio do LaTeX para desenvolver as notações matemáticas necessárias e a *aplicação de um algoritmo*, seguido de comentários que auxiliam na interpretação de cada linha de código incrementada.

Tais estruturas são utilizadas ao longo de todo o *e-book*, visando, portanto, a concisão na teoria abordada e a prática atrelada à utilização de recursos computacionais no auxílio da aprendizagem.

Figura 2 - Parte da versão prévia do *e-book*, no ambiente Jupyter Notebook.**#### Exemplo: distância entre dois pontos**

Calcule a distância entre os pontos $P_1 = (2, 5, 3)$ e $P_2 = (2, -1, 1)$.

Aplicando na fórmula, temos que:

$$d(P_1, P_2) = \sqrt{(2-2)^2 + (-1-5)^2 + (1-3)^2} = \sqrt{40} \approx 6.3246$$

Implementação: distância entre dois pontos

Através do Código `\ref{code:dist-dois-pontos}`, podemos determinar o mesmo resultado obtido no exemplo acima.

In [44]:

```
from math import sqrt # importa a funcao 'sqrt'

p1 = ([2,5,3]) # ponto p1
p2 = ([2,-1,1]) # ponto p2
dist = sqrt((p2[0]-p1[0])**2 + (p2[1]-p1[1])**2 + (p2[2]-p1[2])**2) # distancia
print(round(dist,4)) # resultado com 4 casas decimais
```

6.3246

Note que os resultados coincidem.

Fonte: Autores.

Após o desenvolvimento das seções no ambiente Jupyter, o *e-book* passa pelo processo de conversão, unificando as partes escritas e algorítmicas em um livro digital em formato PDF, como mostra a Figura 3. Observa-se, nesse caso, que há uma transformação visual considerável, no que diz respeito à estrutura, em comparação com a Figura 2. Os títulos das seções recebem numerações que indicam os módulos nos quais os assuntos estão localizados, fazendo com que o aluno obtenha um índice para fácil consulta a todos os assuntos abordados no *e-book*, como seções, subseções, exemplos e implementações. A escrita recebe um realce que difere o título do corpo do texto. O que antes foi desenvolvido com o auxílio do LaTeX, agora toma forma visual e compreensível das notações matemáticas utilizadas, bem como da formatação empregada. No que diz respeito à implementação algorítmica, nota-se que a mesma recebe, a partir da conversão, um plano de fundo em tom cinza, diferenciando-a das demais partes textuais do *e-book*, o que facilita a identificação do código por parte do aluno. Ainda sobre o código, no que diz respeito à sintaxe da linguagem empregada, há um destaque em relação às palavras reservadas da linguagem Python, levando o aluno a conhecer os primeiros diferenciais em relação à mesma. Para dar ao aluno um suporte maior na interpretação do algoritmo, há comentários simples sobre cada comando empregado. Vale salientar que as seções podem se comunicar com outras através de referências, como pode ser observado em “Código 4.4.1.2”. Para seguir até a seção mencionada, basta clicar sobre a numeração, tornando o material mais dinâmico quanto à leitura de seções interdependentes.

Em relação ao acesso, os alunos podem consultar o *e-book* através dos mais variados meios eletrônicos, tais como *smartphones*, notebooks, tablets e computadores. Essa facilidade de acesso se refere ao principal objetivo de desenvolvimento do material, que é o de torná-lo portátil, isto é, acessível em qualquer plataforma e/ou sistema operacional.

Figura 3 - Parte da versão convertida para PDF do *e-book*.

4.4.1.1 Exemplo: distância entre dois pontos Calcule a distância entre os pontos $P_1 = (2, 5, 3)$ e $P_2 = (2, -1, 1)$.

Aplicando na fórmula, temos que:

$$d(P_1, P_2) = \sqrt{(2-2)^2 + (-1-5)^2 + (1-3)^2} = \sqrt{40} \approx 6.3246$$

4.4.1.2 Implementação: distância entre dois pontos Através do Código 4.4.1.2, podemos determinar o mesmo resultado obtido no exemplo acima.

```
1 from math import sqrt # importa a funcao 'sqrt'
2
3 p1 = ([2,5,3]) # ponto p1
4 p2 = ([2,-1,1]) # ponto p2
5 dist = sqrt((p2[0]-p1[0])**2 + (p2[1]-p1[1])**2 + (p2[2]-p1[2])**2)
6 print(round(dist,4)) # resultado com 4 casas decimais
```

Note que os resultados coincidem.

Fonte: Autores.

A Figura 4 mostra o recorte da aplicação algorítmica elencada nas Figuras 2 e 3. Como mencionado anteriormente, o assunto escolhido para demonstrar o processo de desenvolvimento do *e-book* foi o da *distância entre dois pontos*, seguido de um exemplo sob uma ótica puramente matemática, onde o aluno pode aplicar os conceitos adquiridos nas aulas. Após o exemplo, encontra-se a aplicação que envolve o mesmo exemplo matemático, só que agora sob uma ótica algorítmica.

Analisando o código linha a linha, destacam-se os seguintes pontos: importação de bibliotecas que serão necessárias na resolução do exemplo, como a biblioteca *math*, que traz funções matemáticas, tais como raiz quadrada (identificada na Figura 4 como *sqrt*), cálculo de fatorial e determinação do valor absoluto; criação de variáveis, que nada mais são do que objetos que recebem um valor (no exemplo, as variáveis *p1* e *p2* recebem coordenadas de pontos que serão utilizados no cálculo da distância) ou uma expressão (a variável *dist* recebe uma expressão com *sqrt* utilizada para calcular a distância entre dois pontos); retorno do resultado ao usuário, através da função *print* (a função *round*, por exemplo, faz com que o resultado seja retornado com uma certa quantidade de casas decimais. Os símbolos “#” precedem comentários que auxiliam na interpretação do algoritmo por parte do aluno.

No que se refere às principais bibliotecas numéricas na linguagem Python, foram utilizadas ao longo do desenvolvimento do *e-book* a *Numpy*, uma poderosa biblioteca que é usada principalmente para realizar cálculos em arrays multidimensionais. A *Numpy* fornece um grande conjunto de funções e operações de biblioteca que ajudam os programadores a executar facilmente cálculos numéricos. Ela é bastante útil para executar várias tarefas matemáticas como integração numérica, diferenciação, interpolação, extrapolação e muitas outras; a *Scipy*, que foi desenvolvida para trabalhar com rotinas para integração numérica e otimização e a *Matplotlib*, que é uma biblioteca de plotagem de gráficos.

Por fim, os resultados dos códigos implementados e os dos exemplos puramente matemáticos coincidem, acarretando numa melhor formação do aluno frente aos recursos

computacionais disponibilizados na atualidade, recursos estes que, se usados de forma correta e coerente, fazem com que o modo de pensar e agir do aluno sejam multifocais.

Figura 4 - Segmentação do *e-book*, demonstrando um exemplo de código.

```
from math import sqrt # importa a funcao 'sqrt'

p1 = ([2,5,3]) # ponto p1
p2 = ([2,-1,1]) # ponto p2
dist = sqrt((p2[0]-p1[0])**2 + (p2[1]-p1[1])**2 + (p2[2]-p1[2])**2) # distancia
print(round(dist,4)) # resultado com 4 casas decimais
```

Fonte: Autores.

A Figura 5 traz um exemplo de código utilizado na plotagem de gráficos, mais uma utilidade da linguagem Python. Suportada, dentre outras, pelas bibliotecas *Matplotlib* e *Numpy* mencionadas anteriormente, os gráficos gerados em Python podem representar figuras geométricas de maneira dinâmica, como, por exemplo, superfícies cônicas e quádricas, conteúdos vistos ao longo da ementa de Cálculo Vetorial e Geometria Analítica. Tal recurso torna a linguagem Python uma das mais completas, quando se trata de interpretações geométricas de funções de várias variáveis.

Figura 5 – Exemplo de código utilizado para gerar o gráfico de um paraboloide.

```
import numpy as np
import sys
sys.path.append('/anaconda3/lib/python3.6/site-packages')
import plotly.graph_objs as go
from plotly.offline import iplot, init_notebook_mode
init_notebook_mode(connected=True)

x = np.linspace(-5,5,50)
X,Y = np.meshgrid(x,x)
Z = np.sqrt(X**2 + Y**2)
p = go.Surface(z = Z, contours=go.surface.Contours(
    z=go.surface.contours.Z(show=True,
        usecolormap=True,
        highlightcolor="#42f462",
        project=dict(z=True))), name='paraboloide')

data = [p]
sz = 300
layout = go.Layout(title='paraboloide', autosize=False,
    width=sz,height=sz,margin = dict( l = 0,r = 0, b = 0, t = 0))
fig = go.Figure(data=data,layout=layout)
iplot(fig)
```

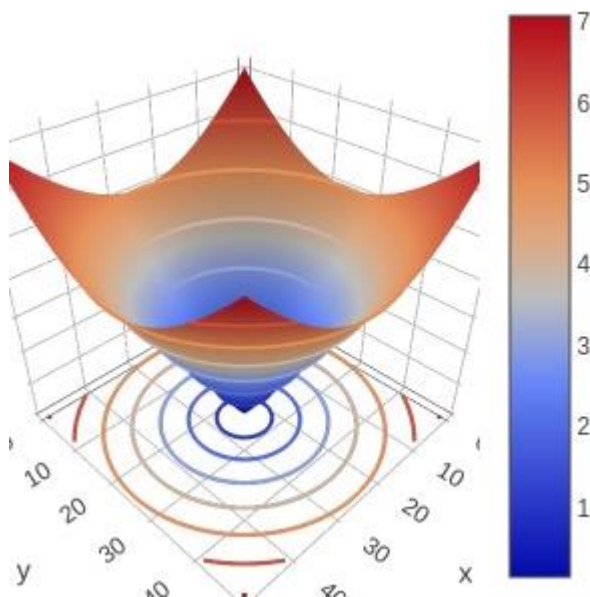
Fonte: Autores.

Como resultado do código exposto na Figura 5, a Figura 6 mostra o gráfico dinâmico de um paraboloide, onde, no ambiente Jupyter, pode-se manipular toda a estrutura gerada, fazendo com que o aluno possa fazer ajustes nas três dimensões. O mesmo ainda pode determinar



gráficos de outras funções apenas manipulando algumas linhas de código; isso faz com que o código seja de fácil reaproveitamento. Desse modo, o aluno, em se tratando de visualização geométrica, tem em mãos tais recursos para uma melhor formação e entendimento dos assuntos ministrados em sala de aula.

Figura 6 – Gráfico de um parabolóide gerado a partir do Python.



Fonte: Autores.

Nota-se, portanto, que todas as etapas mencionadas durante o desenvolvimento do e-book tiveram importância significativa, desde à escrita pré-convertida até a conversão em formatos PDF e EPUB. As múltiplas ferramentas computacionais utilizadas comunicam-se entre si na geração do material final, contribuindo, desse modo, no avanço dos processos de ensino e aprendizagem.

4 CONSIDERAÇÕES FINAIS

O desenvolvimento do *e-book* reflete a inovação do processo ensino-aprendizagem, uma vez que o mesmo quebra o paradigma teórico rebuscado, encontrado na literatura formal, fazendo com que a consulta ao mesmo seja mais direta àquilo que está sendo proposto, no sentido dos assuntos, exemplos e implementações.

A proximidade do tutor com o estudante estimula a utilização de ferramentas computacionais como recursos de suporte ao aprendizado. Estas ferramentas de programação aliadas aos prévios conhecimentos adquiridos pelos tutores das disciplinas criam maiores expectativas de aprovação dos tutorandos mesmo em matérias com altas taxas de reprovação, gerando impactos diretamente nos índices de sucesso dos cursos.

Espera-se que o *e-book* sirva de base para o avanço cognitivo do aluno, no que diz respeito à aplicação do que foi aprendido nos mais variados eixos, tais como nas ciências computacionais e engenharias.

REFERÊNCIAS

CHEUNG, B. et al. **SmartTutor: An intelligent tutoring system in web-based adult education.** Journal of Systems and Software, v. 68, n. 1, p. 11-25, 2003.

HOOSHYAR, D. et al. **SITS: a solution-based intelligent tutoring system for students' acquisition of problem-solving skills in computer programming.** Innovations in Education and Teaching International, v. 55, n. 3, p. 325-335, 2018

KELES, A. et al. **ZOSMAT: Web-based intelligent tutoring system for teaching-learning process.** Expert Systems with Applications, v. 36, n. 2, p. 1229-1239, 2009

KLIMOVA, B. F; POULOVA, P. **Tutor as an important e-learning support.** Procedia Computer Science, v. 3, p. 1485-1489, 2011.

STATIC AND INTERACTIVE E-BOOK DESIGN FOR ANALYTIC GEOMETRY AND VECTOR CALCULUS E-TUTORING IN COMPUTATIONAL SCIENCES AND ENGINEERING COURSES

Abstract: *This work is intended to present static and interactive e-books developed by the Tutorial Education Program - PROTUT at Federal University of Paraíba - UFPB for learning support of Vector Calculus and Analytic Geometry in computer science and engineering undergraduate courses. Tools used for text writing, rendering, plotting and code development are discussed under the concept of "interactive notebook" provided by the Jupyter Project. The interactive platform kernel runs entirely on top of Python 3.x and uses robust Python modules for scientific computing, such as Numpy, Scipy, and Matplotlib.*

Key-words: *E-book. Interactive computing. E-learning. Scientific computing. Analytic Geometry.*