Revista Brasileira de Ensino de Física, vol. 44, e20210239 (2022)
www.scielo.br/rbef
DOI: https://doi.org/10.1590/1806-9126-RBEF-2021-0239

Produtos e Materiais Didáticos

Animações de conceitos da teoria de erros usando Manim/Python

Animations of concepts of the error theory using Manim/Python

Vitor R. Coluci*[1]

[1]Universidade Estadual de Campinas, Faculdade de Tecnologia, Limeira, SP, Brasil.

Recebido em 25 de junho de 2021. Revisado em 09 de novembro de 2021. Aceito em 10 de novembro de 2021.

Animação é o processo de dar movimento a objetos inanimados. Animações têm sido usadas como facilitadoras da aprendizagem tanto em sala de aula como no estudo individual. Diversos *softwares* têm permitido a criação de animações cada vez mais sofisticadas. Particularmente, a biblioteca `Manim` do `Python` permite animar fórmulas matemáticas, figuras geométricas, textos e gráficos. Neste trabalho, apresento o processo usado na criação de animações de conceitos da teoria de erros com o `Manim`. Esses conceitos são geralmente vistos por alunos de Física, Matemática e Engenharia nas disciplinas introdutórias de Física experimental. Por meio das animações, conceitos mostrados em livros de forma estática puderam ser apresentados de uma forma dinâmica. As animações desenvolvidas podem ser usadas em livros e apostilas didáticas interativos e como material complementar para as aulas de laboratório de Física.
Palavras-chave: Animação, Python, Teoria de Erros, Física Experimental.

Animation refers to the process of giving motion to inanimate objects. Animations have been used to facilitate learning both in the classroom and in the individual study. Several softwares have allowed the creation of increasingly sophisticated animations. In particular, `Manim` is a `Python` library that gives motion to mathematical formulas, geometric shapes, texts, and plots. In this work, I present the process used to create animations with `Manim` of concepts of the theory of errors, generally seen by Physics, Mathematics and Engineering students in the introductory Experimental Physics classes. Through animations, concepts originally shown statically in books can be dynamically presented. The animations I present here can be used to produce interactive teaching materials and as complementary material in Physics lab classes.
Keywords: Animation, Python, Theory of Errors, Experimental Physics.

1. Introdução

A série "O Universo Mecânico" [1] possivelmente foi marcante para gerações de professores e alunos. Com um custo de US$ 10 milhões [2], a série resultou de uma revisão dos cursos introdutórios de Física da Caltech, lecionados por Richard Feynman na década de 1960. Criada por David Goodstein na década de 80, a série trata, com uma abordagem histórica, vários temas da Física como Mecânica Clássica, Mecânica Quântica e Eletromagnetismo [3]. A principal forma de ensino na série foi por meio de animações computacionais [2], totalizando aproximadamente 7 $\frac{1}{2}$ h em 550 animações [3]. As animações foram produzidas por Jim Blinn [4] e sua equipe e estão resumidas em 4 vídeos preparados para a conferência anual sobre computação gráfica (ACM Siggraph) [5]. Um registro das animações produzidas por Blinn pode ser visto num relatório detalhado de 1987 [6].

A produção das animações, principalmente aquelas que envolviam manipulações algébricas de fórmulas e equações, seguia o que se chamou de "ballet algébrico" – quando derivações animadas eram feitas de maneira detalhada, porém de forma rápida e que pudesse entreter os telespectadores [3, 6]. Por exemplo, no vídeo sobre as leis de Newton [7], os passos algébricos são animados para se obter a expressão da segunda lei para corpos de massa constante. Essa estratégia captava a atenção e se acreditava que isso tinha relação com a idade dos telespectadores [3]: os jovens, mais acostumados com a televisão, apreciavam mais do que os mais velhos, que se sentiam desconfortáveis com passagens matemáticas que não eram capazes de seguir. Essa tendência parece se manter nos dia de hoje mas com *smartphones* e *tablets* no lugar da antiga televisão de tubo de raios catódicos.

A palavra animação é de origem latina, de *animatio*, que significa ser animado. Deriva da palavra *anima*, que significa *alma* ou *sopro vital*. Assim, o processo de animar é o de dar vida a algo sem vida ou sem movimento. Comparadas às apresentações inanimadas de um tema em um livro tradicional, úteis para informar o conteúdo, as animações são mais apropriadas para explicar os processos da construção do conteúdo. Animações têm se mostrado elementos efetivos para a aprendizagem. Por

* Endereço de correspondência: coluci@unicamp.br

exemplo, Taylor *et al.* mostraram que a apresentação da adição e multiplicação de matrizes é mais efetiva em termos de aprendizagem por meio de animações do que quando feita de forma estática [8]. Nesse trabalho, a adição de duas matrizes é animada fazendo-se os números de cada matriz, em suas respectivas posições, se moverem até a posição na matriz resultante, acrescentando-se o sinal de soma e, finalmente, indicando o resultado.

Animações de fórmulas, gráficos e figuras geométricas como as usadas na série "O Universo Mecânico" tiveram uma grande evolução desde então. Animações mais sofisticadas e mais fáceis de serem produzidas têm sido possíveis com o avanço de ferramentas de animação. Uma dessas ferramentas é o *Mathematical Animation Engine* (`Manim`), uma biblioteca do `Python` [9].

A linguagem `Python` é gratuita, de fácil instalação, bem documentada e conta com muitas bibliotecas disponíveis. Isso tem possibilitado sua aplicação no ensino de vários temas de Física. Ela foi usada em temas mais teóricos como os de Física computacional, para resolver equações diferenciais ordinárias e equações não lineares [10, 11], e de computação quântica, para projetar e executar algoritmos e circuitos quânticos [12]. A interação com objetos e com experimentos virtuais [13–15] e a execução de simulações [16] também tem sido facilitadas com o uso de bibliotecas gráficas para visualização 3D (`VPython`). Já bibliotecas que permitem realizar cálculos numéricos (`numpy`) e fazer gráficos (`matplotlib`) foram usadas para modelar circuitos eletrônicos [17]. A integração do `Python` com computadores e sensores facilitou atividades de Física experimental como, por exemplo, a medição da aceleração da gravidade [18], a análise de vibrações [19] e a criação de interfaces gráficas e manipulação de dados de experimentos envolvendo o plano inclinado, viscosímetro e o pêndulo simples [20], enquanto a biblioteca `uncertainties` foi usada para calcular as incertezas experimentais da tensão e corrente elétrica, empregando-se a Lei de Ohm [21]. A biblioteca `Manim` vem sendo usada para criar conteúdos visualmente atrativos como, por exemplo, a demonstração geométrica do teorema de Pitágoras [22], a explicação de séries de Fourier [23], e muitas outras [24, 25].

Por ser capaz de animar fórmulas, gráficos e objetos geométricos, a biblioteca `Manim` tem facilitado a criação de animações de diversos temas de Física e Matemática. No entanto, até onde foi possível apurar, animações voltadas para temas associados à Física experimental como a definição, análise e tratamento de incertezas experimentais ainda não foram sistematicamente produzidas. Portanto, neste trabalho, apresento uma série de animações curtas produzidas com o `Manim` sobre temas associados à teoria de erros. A teoria de erros tem como objetivos determinar o melhor valor possível para uma grandeza (valor experimental) a partir de medições e determinar sua incerteza [26] e seus conceitos são importantes para os profissionais das áreas de Física, Matemática e Engenharias. A apresentação desses conceitos é geralmente feita no primeiro ano desses cursos, onde conceitos como incerteza, precisão e exatidão, algarismos significativos e distribuição gaussiana são apresentados em disciplinas de laboratórios de Física experimental. Na maioria das vezes, esses conceitos são apresentados e trabalhados juntamente com experimentos de Mecânica. As disciplinas de laboratório foram afetadas fortemente pelo ensino remoto emergencial imposto pela pandemia do COVID-19. Os experimentos tiveram que ser adaptados [27], a avaliação de conteúdos teve que ser repensada [28] e o uso de simulações, laboratórios virtuais [29], *smartphones* e redes sociais se intensificou [30]. Assim, esse trabalho visa fornecer um material complementar para as disciplinas introdutórias de Física experimental e auxiliar no oferecimento dessas disciplinas no formato híbrido.

2. Manim

O `Manim` é uma biblioteca gratuita do `Python`, criada por Grant Sanderson [31], mantedor do projeto *3blue1brown* [24], para desenvolver animações de forma precisa por meio de programação computacional. Como havia pouca documentação para a versão original do `Manim`, algumas pessoas produziram tutoriais e materiais com a descrição da biblioteca[1]. Isso evoluiu para uma comunidade ativa, com quase duzentas pessoas que contribuem com documentação, tutoriais e exemplos [32].

Atualmente, existem três versões do `Manim`: `Manim Community` [32], mantida pela comunidade e que utiliza `pycairo` [33] para renderizar as animações; `Manimcairo` [34], a biblioteca original que também utiliza `pycairo` para criar as animações; e o `ManimGL` [35], que utiliza `openGL` para renderizar os vídeos em tempo real.

Para produzir as animações no `Manim` são usados os conceitos de objeto matemático (`Mobject`), animação e cena. Na maior parte da programação com o `Manim`, o código para produzir a animação usa o método `construct()` de uma classe derivada de `Scene`. Com esse método é possível criar figuras geométricas como retângulos e círculos, fórmulas matemáticas usando a linguagem LaTeX e gráficos 2D e 3D. Várias opções de animação estão disponívies como mostrar/remover, aparecer/desaparecer, girar, mover, escrever, desenhar, dentre outras. A animação de um objeto é incluída usando o comando `play()`. O intervalo entre uma

[1] Veja, por exemplo, o material produzido por Todd A. Zimmerman que contém exemplos e explicações do uso do `Manim` (https://talkingphysics.wordpress.com/2019/01/08/getting-started-animating-with-manim-and-python-3-7/ e o tutorial detalhado produzido por Alexander Vázquez (https://www.youtube.com/watch?v=ENMyFGmq5OA&list=PL2B6OzTsMUrwo4hA3BBfS7ZR34K361Z8F). Os códigos das animações usadas no tutorial estão disponíveis em https://github.com/Elteoremadebeethoven/AnimationsWithManim.

 DOI: https://doi.org/10.1590/1806-9126-RBEF-2021-0239

animação e outra é controlado pelo comando `wait()`. Para ilustrar uma animação com o `Manim`, o código a seguir gera uma animação similar àquela do "O Universo Mecânico" mencionada anteriormente [7]. O resultado pode ser visto em [36].

A animação está contida na classe `Newton` do tipo `Scene`. Inicialmente, os textos e as fórmulas que aparecerão na animação são definidas, juntamente com seus tamanhos e posições iniciais. Os textos são definidos com o objeto `TextMobject` enquanto que as fórmulas são definidas com `TexMobject`. As fórmulas podem ser divididas em partes em uma lista para facilitar as movimentações e transformações. Por exemplo, o objeto matemático `eq_forca` corresponde à $\vec{F} = d\vec{p}/dt$ e foi dividido em 3 partes: ($\vec{F}$ =,d/dt,$\vec{p}$), identificados por `eq_forca[0]`, `eq_forca[1]` e `eq_forca[2]`, respectivamente. Esse tipo de construção permite alterar uma parte específica da fórmula como em `self.play (Transform(momentum_texto,eq_forca[2]))`, onde o conteúdo do texto `momentum_texto` é transformado no conteúdo do terceiro elemento do objeto `eq_forca`, `eq_forca[2]`.

Para a localização e movimentação dos objetos na tela, algumas constantes são usadas como referência, por exemplo, `UP`, `DOWN`, `LEFT`, `RIGHT` correspondem aos vetores (0,1,0), (0,–1,0), (–1,0,0) e (1,0,0), respectivamente. A largura da tela corresponde a 14 unidades e a altura a 8 unidades (razão de aspecto 16:9). Objetos podem ser movidos na tela com comandos como `shift` e `move_to`, indicando a direção e o tamanho do movimento (`shift`) ou a localização da posição final do movimento (`move_to`). As animações das fórmulas usam as opções de aparecimento (`FadeIn`), desaparecimento (`FadeOut`) e de transformação (`Transform`) dentro do comando `play`. Uma espera entre um evento e outro na animação é feita com o comando `wait`, que cria um intervalo de espera de um segundo por padrão. Se o código estiver no arquivo `a.py` então um filme (extensão `.mov`) é obtido com o comando `python3 -m manim a.py Newton -l`, onde `Newton` é o nome da cena que se quer renderizar e a opção `-l` indica que a resolução do vídeo gerado será a de 854×480.

```
from manimlib.imports import *
class Newton(Scene):
  def construct(self):
    # textos
    forca_texto = TextMobject('Força')
    taxa_variacao_texto = TextMobject('Taxa de
     variação')
    momentum_texto = TextMobject('Momentum')
    # tamanhos e posições iniciais
    forca_texto.shift(2*UP).scale(2)
    taxa_variacao_texto.shift(2*UP).scale(2)
    momentum_texto.shift(2*UP).scale(2)
    # fórmulas
    eq_forca = TexMobject(r'\vec{F}=',r'\frac{
    d}{dt}',r'\vec{p}')
    eq_momentum = TexMobject(r'\vec{p}=','m',r
    '\vec{v}')
    eq_a = TexMobject(r'\vec{a}')
    eq_dv_dt = TexMobject(r'\frac{d}{dt} \vec{
    v}')
    # tamanhos e posições iniciais
    eq_forca.scale(2)
    eq_momentum.shift(2*UP).scale(2)
    eq_a.scale(2)
    eq_dv_dt.scale(2).shift(DOWN+2*RIGHT)
    # Faz uma cópia de eq_momentum em eq_mv
    eq_mv = eq_momentum[1:3].copy()
    # Inicio das animações
    # Faz aparecer a palavra "Força"
    self.play(FadeIn(forca_texto))
    # aguarda 1s
    self.wait()
    # Transforma "Força" em $\vec{F}$
    self.play(Transform(forca_texto, eq_forca
    [0]))
    self.wait()
    # Faz aparecer a palavra "taxa de variação
    "
    self.play(FadeIn(taxa_variacao_texto))
    self.wait()
    # Transforma "taxa de variação" em $d/dt$
    self.play(Transform(taxa_variacao_texto,
    eq_forca[1]))
    self.wait()
    # Faz aparecer a palavra "Momentum"
    self.play(FadeIn(momentum_texto))
    self.wait()
    # Transforma "Momentum" em  $\vec{p}$
    self.play(Transform(momentum_texto,
    eq_forca[2]))
    self.wait()
    # Remove os textos
    self.remove(forca_texto,
    taxa_variacao_texto,momentum_texto)
    # Movimenta $\vec{F} = d\vec{p}/dt$ uma unidade para
    baixo
    self.play(eq_forca.shift, DOWN)
    self.wait()
    # Faz aparecer $\vec{p} = m\vec{v}$
    self.play(FadeIn(eq_momentum))
    self.wait()
    # Move $m\vec{v}$ no lugar de $\vec{p}$ e faz desaparecer
     $\vec{p}$
    self.play(eq_mv.move_to,eq_forca[2].
    get_center()+0.1*UP+0.4*RIGHT,FadeOut(
    eq_forca[2]))
    self.wait()
    # Movimenta $m$ para fora de $d/dt$
    self.play(eq_mv[0].shift,LEFT,eq_forca[1].
    shift,0.92*RIGHT,eq_mv[1].shift,0.1*RIGHT)
    self.wait()
    # Mostra na tela $d/dt$ que ficará no lugar
    de $d/dt$ da eq_forca
    self.add(eq_dv_dt)
    # Remove da tela $d/dt$ da eq_forca e $m$ da
    eq_mv
    self.remove(eq_forca[1], eq_mv[1])
    # Transforma dv/dt em a
    self.play(Transform(eq_dv_dt,eq_a.move_to
    ((eq_dv_dt.get_center()+0.5*LEFT))))
```

3. Desenvolvimento das Animações

Uma série com nove animações foi desenvolvida sobre temas que são tradicionalmente abordados na primeira disciplina de Física experimental: método científico,

Tabela 1: Vídeos das animações produzidas.

Tema (duração)
Método Científico (1m:32s)
Precisão e Exatidão (1m:03s)
Distribuição de medições (1m:11s)
Distribuição gaussiana (1m:26s)
Erro padrão (3m:45s)
Algarismos significativos (4m:55s)
Representação de uma grandeza (2m:01s)
Método dos mínimos quadrados (3m:13s)
Propagação de incertezas (2m:58s)

precisão e exatidão, distribuição de medições de grandezas experimentais, distribuição gaussiana, desvio padrão do valor médio (erro padrão), algarismos significativos, representação de uma grandeza experimental, método dos mínimos quadrados e propagação de incertezas. As animações foram produzidas com a versão original do `Manim` [34]. Os códigos-fonte das animações estão disponíveis no repositório Github [37] e os vídeos das animações podem ser vistos no YouTube [38]. Os vídeos da série estão indicados na Tabela 1.

As animações compõem o acervo digital do projeto *Explora* [39] da Faculdade de Tecnologia da UNICAMP. O projeto conta também com um espaço com experimentos e demonstrações para visitação e complementação de aulas teóricas e práticas de Física e Matemática. Além disso, ele conta com um grupo de alunos que estão trabalhando no desenvolvimento de animações com o `Manim`.

Na primeira animação da série são apresentadas as partes do método científico como a indicação do fenômeno a ser estudado, a proposição de experimentos para se realizar medições e obter dados sobre o fenômeno, a formulação de modelo e o confronto entre modelo e as leis científicas. A apresentação do método científico foi animado por meio de retângulos (`Rectangle`), que continham as partes do método e que iam aparecendo na tela (`self.play(FadeIn (caixa_experimento))`, e por setas (`Arrow`), que indicavam a sequência de aplicação do método. Textos curtos foram incorporados à animação para uma explicação sucinta das partes do método (`TextMobject("Reprodução do fenômeno")`). É uma animação simples mas que pode auxiliar o professor em mostrar a sequência de aplicação do método científico.

Os conceitos de precisão e exatidão são ilustrados na segunda animação, onde são apresentadas quatro situações da distribuição de um conjunto de medições sob as mesmas condições. As situações das distribuições quando o valor da grandeza experimental obtido é preciso e exato; preciso e inexato; impreciso e exato; e impreciso e inexato. Esses casos foram animados usando a figura de um alvo de tiro, cujo centro indica o valor verdadeiro da grandeza física e os tiros indicam as medições realizadas da grandeza experimental. Além de textos curtos, os objetos geométricos círculo (`Circle`), segmento de reta (`Line`), seta e retângulo foram os elementos usados na animação. Para indicar a dispersão das medições, as posições dos "tiros" no alvo foram geradas seguindo uma distribuição normal (`random.randn`).

A distribuição de medições de grandezas experimentais quando realizadas sob as mesmas condições foi o tema da terceira animação da série. Os valores das medições são representados por pequenos segmentos de reta localizados ao longo do eixo horizontal, de forma similar a usada por Vuolo (pag. 80) [26]. A animação ilustra como essas medições vão se distribuindo num histograma para conjuntos com 50 a 500 medições (com incremento de 50). Juntamente ao histograma do conjunto de 500 medições, a função gaussiana é representada no gráfico. Além dos objetos textuais, essa animação usou o objeto gráfico (`self.setup_axes(animate=False)`) para apresentar o histograma e a função gaussiana (`func1 = self.get_graph(self.f1)` e `self.play(FadeIn (func1))`). O histograma foi construído "manualmente". A partir da distribuição gaussiana das medições, os valores do histograma foram calculados e usados para se criar retângulos. Após isso, os retângulos foram posicionados no gráfico para compor o histograma.

Após a apresentação da função gaussiana na terceira animação, detalhes dessa função foram mostrados na quarta animação. O gráfico da gaussiana é apresentado na tela e, ao lado dele, algumas informações sobre a função são informadas de forma sequencial. Primeiro, é mostrada a fórmula da função usando o objeto matemático `TexMobject`. Em seguida, é indicado que a função é simétrica e que a integral dela sobre todo o eixo x vale 1. Depois disso, a localização da média μ é indicada no gráfico e que, para $x = \mu$, a função é máxima. Em seguida, informa-se que σ é o desvio padrão. Finalmente, as áreas abaixo da função com os respectivos valores da integral são mostradas para os intervalos $\mu \pm \sigma$, $\mu \pm 2\sigma$ e $\mu \pm 3\sigma$ (Fig. 1). Para isso, foi usado os comandos `area = self.get_area(f,a,b)`

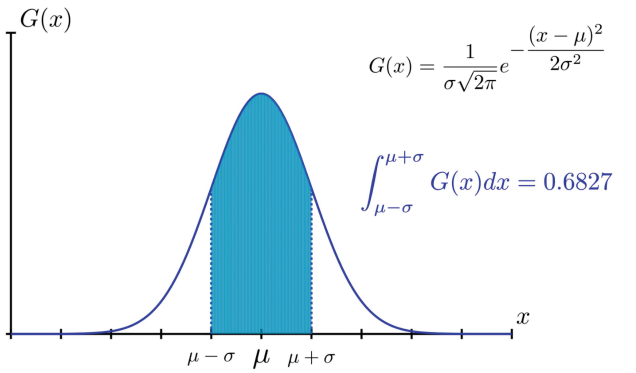

Figura 1: Cena da animação sobre a função gaussiana. A área sob a função e o valor da integral para o intervalo $\mu \pm \sigma$ são indicados na tela.

 DOI: https://doi.org/10.1590/1806-9126-RBEF-2021-0239

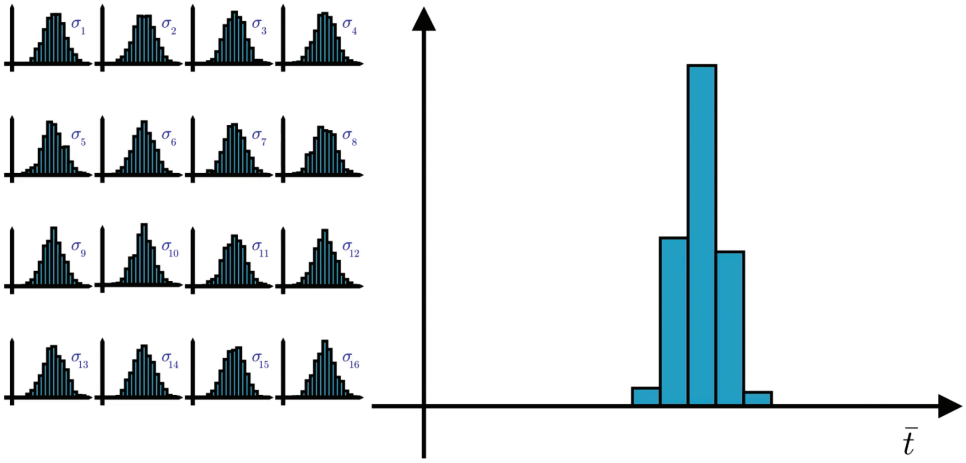

Figura 2: Cena da animação sobre o desvio padrão da média. À direita é mostrado o histograma dos valores médios das distribuições de medições mostradas à esquerda.

para se obter a área da função f entre os pontos a e b e o comando `self.play(ShowCreation(area))` para exibir a área sob a função no intervalo especificado.

O conceito do desvio padrão da média – erro padrão – é ilustrado na quinta animação. A animação se inicia com um histograma de um conjunto de N medições do tempo de queda para um experimento de queda livre, onde são indicados o valor médio $\bar{t}$ e o desvio padrão σ do conjunto. Em seguida, simula-se que o experimento é repetido k vezes sob as mesmas condições, obtendo k novos histogramas (k =16 no caso). Esses histogramas vão sendo representados na tela assim que são construídos. Mostrar o conjunto do histogramas permite passar a ideia, de uma forma visual, de como será feito o cálculo do erro padrão. Sobre cada histograma gerado é indicado o valor de $\bar{t}$ para aquele conjunto de medições. Um histograma para esses valores de $\bar{t}$ é então construído e apresentado na tela (Fig. 2). Em seguida, é apresentada a expressão para o desvio padrão da média

$$\sigma_m^2 = \frac{1}{k}\sum_{j=1}^{k}(\bar{t_j} - t_{mv})^2 \qquad (1)$$

onde t_{mv} é o valor médio verdadeiro. A partir dessa expressão, várias manipulações algébricas ([26] pag. 99–100) são animadas para se obter a expressão aproximada do desvio padrão da média $\sigma_m \simeq \sigma/\sqrt{N}$. A animação termina com a apresentação do gráfico de σ_m/σ em função de N para ilustrar como o desvio padrão da média decai com o aumento do número de medições.

A sexta animação foi dedicada ao conceito de algarismos significativos. A animação começa apresentando o número 0.0000314159265359 para mostrar como proceder com zeros à esquerda do primeiro algarismo não-nulo. Em seguida, é mostrado como se determinam quais são os algarismos significativos e quais são os que não têm significado, a partir da incerteza padrão, para o caso de $\bar{y} = 3.141592$ e $\sigma = 0.031473$. Isso é feito para o número 3.14159 onde a probabilidade de ocorrer os

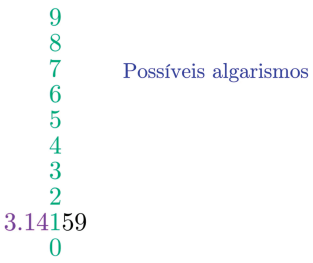

Figura 3: Cena da animação sobre algarismos significativos. Os algarismos significativos são indicados em verde. Em vermelho, estão indicados os possíveis algarismos para um certo dígito do número. Em branco, são os dígitos que ainda não foram analisados na animação.

algarismos possíveis (0 a 9) de um dígito do número é representada pelo tamanho do algarismo. Usar o tamanho do algarismo para expressar sua probabilidade de ocorrer foi a forma lúdica adotada para atrair a atenção e relacionar isso com o atributo de ser significativo ou não. Se a probabilidade para um algarismo for maior que a dos outros, ele é significativo. Por outro lado, se a probabilidade for aproximadamente igual a dos outros, ele não é signficativo. Da esquerda para a direita, os algarismos do número 3.14159 vão sendo analisados e, após a análise, ele recebe a cor verde, se for significativo, ou a cor vermelha, se não o for (Fig. 3). Ao final, apresenta-se a quantidade de algarismos significativos que deve ser usada para representar o desvio padrão. Isso é feito por meio da análise da fórmula para o desvio padrão do desvio padrão de cada medição $\sigma_{\sigma^2} = \sigma[2/(N-1)]^{\frac{1}{4}}$ [40], onde se calcula a quantidade de medições (N) necessárias para que se tenha σ_{σ^2} que afete a terceira casa decimal após o ponto em $\sigma = 0.031473$, ou seja, $\sigma_{\sigma^2} = 0.001$. O valor de N de 2.5 milhões é obtido, indicando que apenas um algarismo significativo deve ser geralmente usado para representar o desvio padrão.

A sétima animação mostra como representar uma grandeza experimental quando apenas erros aleatórios

estão presentes, ou seja, usando a média e o desvio padrão da média do conjunto de medições. A animação mostra os cálculos para um conjunto de 11 medições referentes ao alcance de um projétil. Ao final, os valores da média e do desvio padrão da média são representados propositalmente com muitos algarismos para que fosse indicada a conversão desses valores de forma a ter a quantidade correta de algarismos significativos, como apresentado na animação anterior.

A penúltima animação da série apresenta o método dos mínimos quadrados, exemplificando-o para o caso do ajuste linear. A animação começa apresentando um gráfico com 4 pontos experimentais. Em seguida, é informado que cada ponto tem associado a ele uma incerteza. Para se representar a incerteza, indicou-se, próximo ao ponto, vários outros pontos cuja intensidade da cor variava de acordo com uma distribuição normal. Isso gerava um efeito de um "borrão" próximo ao ponto, o qual indicava que o valor daquele ponto era incerto. Sobre esses "borrões", foram construídas as barras de erros na cena seguinte. Em seguida, algumas manipulações algébricas foram animadas tendo como ponto de partida a probabilidade de obter o valor (x_i, y_i, σ_i):

$$P_i \propto \frac{1}{\sigma_i} \exp\left[-\frac{1}{2}\left(\frac{y_i - y_{teo}(a,b)}{\sigma_i}\right)^2\right] \qquad (2)$$

onde $y_{teo}(a,b) = a + bx$ é a função que se quer ajustar ao conjunto de pontos, e tendo como ponto de chegada a probabilidade de obter *todos* os pontos do conjunto

$$P \propto \frac{1}{\sigma_1 \sigma_2 \ldots \sigma_n} \exp\left[-\frac{1}{2} S^2\right]. \qquad (3)$$

Nessa última expressão, $S^2 = \sum_{i=1}^{N} \left(\frac{y_i - y_{\text{teo}}(a,b)}{\sigma_i}\right)^2$ é a quantidade que deve ser minimizada a fim de se obter os melhores parâmetros a e b que maximizam P (método da máxima verossimilhança [26]). Nesse momento da animação, o gráfico com os pontos experimentais volta à cena e quadrados de lado $y_i - y_{\text{teo}}(a,b)$ são representados próximos aos respectivos pontos para ilustrar o conceito dos "mínimos quadrados" (Fig. 4). Neste ponto da animação, o coeficiente da reta a ser ajustada é variado e o respectivo efeito nos quadrados é mostrado. Com isso, passa-se a ideia de que devem existir valores ótimos de a e b que levem a um menor valor da soma da área dos quadrados. A animação então segue com a dedução desses melhores valores de a e b por meio da minimização de S. Finalmente, usando a notação $[f] = \sum f_i/\sigma_i^2$ e $<f> = [f]/[1]$ [41], os valores de a e b são expressos como

$$b = \frac{[1][xy] - [x][y]}{[1][x^2] - [x][x]} \qquad (4)$$

e

$$a = < y > - b < x > . \qquad (5)$$

Finalmente, a última animação da série apresenta o conceito de propagação de incertezas. A partir de uma

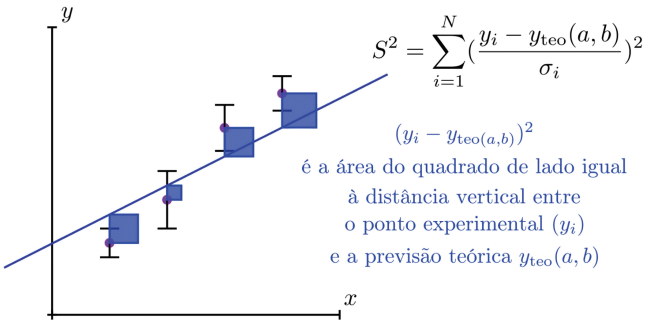

Figura 4: Cena da animação sobre o método dos mínimos quadrados. Na animação, os coeficientes da reta são alterados levando a um movimento da reta no gráfico e a mudanças nos tamanhos dos quadrados amarelos.

grandeza física w que depende de outras grandezas, $w = f(x, y, z, \ldots)$, a animação mostra alguns casos do cálculo da incerteza em w feito a partir de

$$\sigma_w^2 = \left(\frac{\partial w}{\partial x}\right)^2 \sigma_x^2 + \left(\frac{\partial w}{\partial y}\right)^2 \sigma_y^2 + \left(\frac{\partial w}{\partial z}\right)^2 \sigma_z^2 + \ldots \qquad (6)$$

Os casos ilustrados foram $w = x+y+z$, $w = ax+by+cz$, $w = xyz$ e $w = x^p y^q z^r$. Para cada um deles, o processo de derivação de w com relação às váriaveis x, y e z é animado até se chegar a expressão final para σ_w^2. Ao final, o cálculo da propagação de incerteza é ilustrado para o caso do volume de um cilindro $V = \pi R^2 L$ para $L = (103 \pm 2)$ cm e $R = (15.2 \pm 0.5)$ cm.

4. Considerações finais

A criação de animações com o `Manim` requer um conhecimento de programação em `Python`. Isso pode ser uma barreira para algumas pessoas, mas que pode ser vencida com algumas horas de estudo a partir dos vários bons materiais sobre `Python` disponíveis (veja por exemplo [42]). Por outro lado, a linguagem `Python` é muito popular e ensinada em vários cursos de graduação. Isso permite atrair alunos e formar uma equipe para produzir as animações. Além da programação, o desenvolvimento do roteiro da animação tem um papel central. Isso envolve, por exemplo, decidir como abordar um determinado tema, quais objetos gráficos usar e que tipo e quantidade de manipulações algébricas serão animadas. É principalmente neste ponto que os professores têm muito a contribuir, dada a experiência de ter lecionado inúmeras vezes os temas a serem animados. A fim de facilitar a aprendizagem tanto do `Python` como do `Manim`, elaboramos um manual introdutório [43] sobre eles, abordando os principais tópicos para se iniciar a criação das animações. O manual apresenta o `Manim` na versão mais nova (`Manim Community`) numa plataforma colaborativa, facilitando assim a instalação da biblioteca e o desenvolvimento dos códigos.

As animações desenvolvidas neste trabalho podem ser integradas em ambientes virtuais de aprendizagem

 DOI: https://doi.org/10.1590/1806-9126-RBEF-2021-0239

(Moodle, por exemplo) e serem usadas como material complementar de estudo. Exercícios de fixação podem ser criados usando as animações em conteúdos de HTML5 interativo por meio do H5P [44]. Por exemplo, no conteúdo de vídeo interativo [45], diferentes tipos de questões podem ser feitas na própria tela, durante a exibição do vídeo. Finalmente, ao utilizar também simulações interativas [46–48], é possível produzir materiais dinâmicos e interativos no estilo do livro de Cálculo de Briggs, Cochran, Gillett e Schulz [49], onde os leitores podem interagir com os gráficos lá apresentados.

Pretendemos usar o `Manim` também para desenvolver animações de conteúdos de Matemática do ensino médio a fim de aumentar o engajamento e motivação dos alunos em temas das Ciências Exatas. Além disso, cursos de extensão estão sendo preparados para serem oferecidos a professores do ensino médio para capacitá-los no desenvolvimento de animações facilitadoras de aprendizagem.

Agradecimentos

Agradeço a Eric S. S. Kishimoto por simplificar o código da animação da segunda lei de Newton para massa constante e por elaborar o manual sobre `Manim/Python`.

Referências

[1] https://www.youtube.com/playlist?list=PL8_xPU5epJddRABXqJ5h5G0dk-XGtA5cZ, acessado em 05/11/2021.

[2] D. Goodstein, Journal of Science Education and Technology **1**, 149 (1992).

[3] D.L. Goodstein e R.P. Olenick, American Journal of Physics **56**, 779 (1988).

[4] https://www.jimblinn.com/, acessado em 05/11/2021.

[5] https://youtu.be/LqOiR4cSstM, acessado em 05/11/2021.

[6] J.F. Blinn, em: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive techniques* (ACM SIGGRAPH and IEEE Technical Committee on Computer Graphics, Anaheim, 1987).

[7] https://www.youtube.com/watch?v=Zawxywc3Stc&list=PL_CWSYH2QtJNr0j01_36rfPHqHs8HMQ_-&index=8, acessado em 05/11/2021.

[8] M. Aylor, D. Pountney e I. Malabar, Journal of Further and Higher Education **31**, 249 (2007).

[9] https://www.python.org/, acessado em 05/11/2021.

[10] A. Backer, Computing in Science & Engineering **9**, 30 (2007).

[11] P.H. Borcherds, Computer Physics Communications **177**, 199 (2007).

[12] G.F. de Jesus, M.H.F. da Silva, T.G. Dourado Netto, L.Q. Galvão, F.G. de Oliveira Souza e C. Cruz, Revista Brasileira de Ensino de Física **43**, e20210033 (2021).

[13] D. Scherer, P. Dubois e B. Sherwood, Computing in Science & Engineering **2**, 56 (2000).

[14] J. Lincoln, The Physics Teacher **57**, 60 (2019).

[15] K.L. Cristiano, A. Estupiñán e D.A. Triana, Journal of Physics: Conference Series **1247**, 012044 (2019).

[16] P. Freitas-Lemes, D.C. Vilela, M.G. Guarnieri, R. Oliveira Prado, T.F. Medeiros e J.S.E. Germano, Revista Brasileira de Ensino de Física **41**, e20180090 (2019).

[17] L. Gojković, S. Malijević e S. Armaković, Physics Education **55**, 055016 (2020).

[18] A. Martínez, C. Nieves e A. Rúa, The Physics Teacher **59**, 134 (2021).

[19] G.H.C. Barbosa, M. Varanis, K.M.S. Delgado e C. Oliveira, Revista Brasileira de Ensino de Física **42**, e20200167 (2020).

[20] J.T. Carvalho Neto, F.R. Apolinário e A.A. Soares, Revista Brasileira de Ensino de Física **40**, e1504 (2018).

[21] F.M. Couto e J.F. Silva, Revista Brasileira de Ensino de Física **43**, e20200415 (2021).

[22] https://youtu.be/14FC6mIRyNQ, acessado em 05/11/2021.

[23] https://www.youtube.com/watch?v=r6sGWTCMz2k&list=PLZHQObOWTQDNPOjrT6KVlfJuKtYTftqH6&index=4, acessado em 05/11/2021.

[24] https://www.3blue1brown.com, acessado em 05/11/2021.

[25] https://www.reddit.com/r/manim/, acessado em 05/11/2021.

[26] J.H. Vuolo, *Fundamentos da Teoria de Erros* (Editora Blucher, São Paulo, 1996), 2^a ed.

[27] M.F.J. Fox, A. Werth, J.R. Hoehn e H.J. Lewandowski, arXiv:2007.01271 (2020).

[28] D. Gende, The Physics Teacher **58**, 440 (2020).

[29] J. Lincoln, The Physics Teacher **58**, 444 (2020).

[30] D.J. O'Brien, American Journal of Physics **89**, 403 (2021).

[31] https://www.3blue1brown.com/about, acessado em 05/11/2021.

[32] Manim Community, Mathematical Animation Framework, v0.12.0 (2021), disponível em https://www.manim.community/, acessado em 5/11/2021.

[33] https://pypi.org/project/pycairo/, acessado em 05/11/2021.

[34] https://github.com/3b1b/manim, acessado em 05/11/2021.

[35] https://pypi.org/project/manimgl/, acessado em 05/11/2021.

[36] https://github.com/vitorcoluci/teoria-erros-manim/blob/main/newton.mp4, acessado em 05/11/2021.

[37] https://github.com/vitorcoluci/teoria-erros-manim, acessado em 05/11/2021.

[38] https://www.youtube.com/playlist?list=PLw-0KOdAebch67Q3B7QiuNUrn27lLFhtB, acessado em 05/11/2021.

[39] https://wordpress.ft.unicamp.br/explora/, acessado em 05/11/2021.

[40] G.L. Squires, *Practical Physics* (Cambridge University Press, Cambridge, 2001), 4^a ed.

[41] L. Lyons, *A Practical Guide to Data Analysis for Physical Science Students* (Cambridge University Press, Cambridge, 1991), 1^a ed.

[42] H. Fangohr, *An Introduction to Python for Computational Science and Engineering* (2021), https://fangohr.github.io/introduction-to-python-for-computational-science-and-engineering/, acessado em 05/11/2021.
[43] https://github.com/vitorcoluci/teoria-erros-manim/blob/main/Manual_Manim_Github.ipynb, acessado em 05/11/2021.
[44] https://h5p.org/, acessado em 05/11/2021.
[45] https://h5p.org/interactive-video, acessado em 05/11/2021.
[46] https://ophysics.com/, acessado em 05/11/2021.
[47] https://vascak.cz/physicsanimations.php?l=en, acessado em 05/11/2021.
[48] https://phet.colorado.edu/, acessado em 05/11/2021.
[49] W.L. Briggs, L. Cochran, B. Gillett e E. Schulz, *Calculus, Early Transcendentals* (Pearson, New York, 2019), 3^a ed.