Revista Eletrônica Paulista de Matemática

ISSN 2316-9664
v. 23, n. 1, jul. 2023
Artigo de Iniciação Científica

Leonardo Hannas de Carvalho Santos
ICMC/EESC
Universidade de São Paulo
leonardohannas@usp.br

Equações diferenciais impulsivas: uma abordagem sobre estabilidade e métodos numéricos

Impulsive differential equations: an approach on stability and numerical methods

Resumo

O artigo pode ser dividido em três partes:
→ Equações Diferenciais Impulsivas (ou simplesmente EDIs): conceitos fundamentais, como a descrição de sistemas com impulsos pré-estabelecidos, a existência e continuação de soluções no intervalo (ou no espaço vetorial) de análise, a dependência de valores iniciais, além da abordagem de alguns exemplos de aplicação prática.
→ Estabilidade de soluções de EDIs: são definidos os tipos de estabilidade, além dos seus respectivos critérios. Também é apresentado o Teorema de Lyapunov, o qual analisa a estabilidade de uma solução partindo-se da definição de funções de energia.
→ Métodos numéricos para a resolução de EDIs: são analisados alguns métodos computacionais para a resolução de problemas desta natureza. Por exemplo, os métodos de passo constante, como os métodos de Runge-Kutta.
Palavras-chave: EDIs. Impulsos. Lyapunov. Método de Runge-Kutta.

Abstract

This paper can be divided in three parts:
→ Impulsive Differential Equations (or simply IDEs): fundamental concepts, such as the description of systems with pre-established impulses, the existence and continuity of solutions in the interval (or in the vectorial space) analyzed, the initial value dependence and the approach of some practical examples.
→ Stability of solutions of IDEs: the types of stability are defined as well as their criteria. In addition, the Lyapunov's Theorem is presented in order to analyse the stability of a given solution, initiating by the definition of energy's functions.
→ Numerical methods for solving IDEs: some of the existing methods are analysed. For example, the step constant methods such as the Runge-Kutta methods.
Keywords: IDEs. Impulses. Lyapunov. Runge-Kutta Method.

Artigo recebido em set. 2022 e aceito em fev. 2023

1 Introdução

Sistemas impulsivos estão presentes em diversos ramos da ciência e são bastante úteis para a modelagem matemática de sistemas reais. Tal variedade pode ser verificada analisando-se alguns exemplos de diferentes áreas, como o aumento impulsivo de uma população de peixes, modelos impulsivos em redes neurais, ou ainda através da flutuação de preços através de impulsos. A seguir, serão analisados um pouco mais detalhadamente os modelamentos matemáticos de cada um dos exemplos citados, através da utilização de efeitos impulsivos.

1.1 Aumento impulsivo de uma população de peixes

Considerando-se uma população de peixes homogênea num lago que liga duas porções de um riacho, o comportamento de tal população é descrito da seguinte maneira

$$\dot{N}(t) = N \times F(N) + u \tag{1}$$

em que $N(t)$ é o tamanho da população no instante t, $N(t) \times F(N(t))$ é a taxa natural de crescimento da população e $u \geq 0$ é o fluxo constante de peixes do riacho para o lago (LIU, 1995 apud STAMOVA; STAMOV, 2016, p. 2).

A equação (1) não considera efeitos bruscos sobre o número de indivíduos da população em questão. Efeitos como o envenenamento de indivíduos por contaminação do lago, ou o chegar de uma ninhada deverão ser considerados. Com isso, X. Liu propõe que nos instantes de tempo t_k, $k \in \mathbb{N}^*$, ocorrem ninhadas, que serão interpretadas como sendo aumentos impulsivos da população de peixes. Logo, da equação (1), tem-se o sistema (2):

$$\begin{cases} \dot{N}(t) = N \times F(N) + u, \ t \neq t_k, \ t \geq 0 \\ \Delta N(t_k) = N(t_k^+) - N(t_k^-) = I_k(N(t_k)), \ k \in \mathbb{N}^* \end{cases} \tag{2}$$

em que $N(t_k^-) = N(t_k)$ e $N(t_k^+)$ são as populações antes e depois do impulso, e $I_k \in \mathbb{R}$ são funções que caracterizam a magnitude do impulso no instante t_k. Obviamente, se $I_k > 0$, então há um aumento da população e, se $I_k < 0$, tem-se sua diminuição. Pode-se mostrar, por fim, que a existência dos impulsos pode configurar estabilidade à população de peixes.

1.2 Modelos impulsivos de redes neurais

Num contexto geral redes neurais são eficientes na identificação de padrões e na previsão de dados. Em 1988, Chia e Yang propuseram uma nova classe de sistemas de processamento de informações, as *CNN's - "Cellular Neural Networks"* (ou *"Redes Neurais Celulares"*). Tais redes neurais são aplicadas em programação linear e não-linear, otimização, reconhecimento de padrões, visão computacional, etc. As equações a seguir descrevem uma *"Hopfield-type CNN"*

$$\dot{x_i}(t) = -c_i x_i(t) + \sum_{j=1}^{n} a_{ij} f_j(x_j(t)) + I_i$$

ou, considerando-se um atraso $\tau(t)$, tem-se

$$\dot{x}_i(t) = -c_i x_i(t) + \sum_{j=1}^{n} a_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} b_{ij} f_j(x_j(t - \tau_j(t))) + I_i$$

em que $i \in \mathbb{N}^*$ corresponde ao índice da unidade na rede neural, $x_i(t)$ corresponde ao estado da i-ésima unidade no instante t, $f_j(x_j(t))$ denota a saída da j-ésima unidade no instante t, a_{ij} refere-se à força (ou peso de ponderação) da j-ésima unidade sobre a i-ésima unidade no tempo t. Analogamente, b_{ij} corresponde à força da unidade x_j sobre x_i no instante de tempo $t - \tau_j(t)$, I_i é a tendência de comportamento externa sobre x_i, $\tau_j(t)$ caracteriza o atraso na transmissão da informação na j-ésima unidade, com $0 \leq \tau_j(t) \leq \tau = constante$. Por fim, c_i representa a taxa na qual a i-ésima unidade atinge o seu potencial de descanso, quando desconectada da rede e isolada de interferências externas.

Adicionando-se perturbações instantâneas sobre as unidades x_i nos instantes de tempo t_k $(k \in \mathbb{N}^*)$ tem-se os seguintes sistemas

$$\begin{cases} \dot{x}_i(t) = -c_i x_i(t) + \sum_{j=1}^{n} a_{ij} f_j(x_j(t)) + I_i, \ t \neq t_k, \ t \geq 0 \\ \Delta x_i(t) = x_i(t_k^+) - x_i(t_k) = P_{ik}(x_i(t_k)), \ k \in \mathbb{N}^* \end{cases} \tag{3}$$

ou, para modelos impulsivos de *CNN* com atrasos, a primeira equação de (3) será dada por

$$\dot{x}_i(t) = -c_i x_i(t) + \sum_{j=1}^{n} a_{ij} f_j(x_j(t)) + \sum_{j=1}^{n} b_{ij} f_j(x_j(t - \tau_j(t))) + I_i, \quad t \neq t_k, \ t \geq 0$$

em que t_k são os instantes de tempo referentes às perturbações impulsivas e $P_{ik}(x_i(t_k))$ são as variações abruptas no estado $x_i(t)$ em $t = t_k$.

1.3 Modelo impulsivo de flutuação de preços

Sendo $p(t)$ o preço de um produto qualquer no mercado, a seguinte equação foi proposta

$$\frac{1}{p}\frac{dp}{dt} = F(D(p_d), S(p_s)), \ t \geq 0 \tag{4}$$

em que D e S são, respectivamente, a demanda e a oferta (*"supply"*) do produto em questão (MACKEY; BELAIR, 1989 apud STAMOVA; STAMOV, 2016, p. 4).

A função de variação de preço $F(D, S)$ satisfaz às seguintes condições

$$\begin{cases} F(D,S) = 0 \iff D = S \\ \frac{dF}{dD} \geq 0, \ \frac{dF}{dS} \leq 0 \ . \end{cases}$$

Um caso simples pode ser dado por: $F(D, S) = D - S$.

Finalmente, adicionando-se variações abruptas de preço na equação (4), tem-se o seguinte sistema:

$$\begin{cases} \frac{\dot{p}(t)}{p(t)} = F\left(D\left(p(t)\right), S\left(p(t)\right)\right), \ t \neq t_k \\ \Delta p(t_k) = p(t_k^+) - p(t_k) = P_k\left(p(t_k)\right), \ k \in \mathbb{N}^* \ . \end{cases}$$

Após a apresentação de algumas possíveis aplicações de modelagem através de sistemas impulsivos, as próximas seções abordarão, respectivamente, a teoria básica das EDIs, as condições de estabilidade e fronteira, as Funções de Lyapunov e o Teorema de Estabilidade de Lyapunov. Finalmente, serão tratados os métodos numéricos de Runge-Kutta para a resolução de EDOs e, a seguir, será proposto um algoritmo computacional, visando à resolução de EDIs.

2 Teoria Básica - Equações Diferenciais Ordinárias Impulsivas

Considerando-se o seguinte sistema, denota-se por $x(t) = x(t; t_0, x_0)$ a solução de (5), satisfazendo-se a condição inicial $x(t_0^+; t_0, x_0) = x_0$.

$$\begin{cases} \dot{x}(t) = f(t, x), \ t \neq \tau_k(x(t)) \\ \Delta x(t) = I_k(x(t)), \ t = \tau_k(x(t)), \ k \in \mathbb{Z}^* \end{cases} \tag{5}$$

Os pontos onde se verifica a existência de impulsos são definidos pelos conjuntos σ_k, dados por

$$\sigma_k = \{(t, x) | t = \tau_k(x), x \in \Omega\} \ \text{(Hipersuperfícies)}.$$

Além disso, denota-se a seguinte convenção:

$$x(t_k^-) = x(t_k) \ \ e \ \ x(t_k^+) = x(t_k) + I_k(x(t_k)),$$

sendo $x(t_k^-)$ e $x(t_k^+)$ os limites laterais à esquerda e à direita, respectivamente.

Em seguida, como, usualmente, a variável t designa tempo, a seguinte relação de ordem é assumida como sendo válida:

$$\tau_k < \tau_{k+1}(x) \ \ e \ \lim_{k \to \pm\infty} \tau_k(x) = \pm\infty, \ x \ \in \ \Omega$$

Por fim, assume-se que cada solução $x(t)$ intersecta cada hipersuperfície σ_k em, no máximo, uma única vez. Partindo-se dessa hipótese, verifica-se a ausência do fenômeno de batimento e o sistema (5) é reduzido a

$$\begin{cases} \dot{x}(t) = f(t, x), \ t \neq t_k \\ \Delta x(t) = I_k(x(t)), \ t = t_k, \ k \in \mathbb{Z}^* \end{cases} \tag{6}$$

em que os impulsos ocorrem em $t_k < t_{k+1} (k \in \mathbb{Z}^*)$ e $\lim_{k \to \pm\infty} t_k = \pm\infty$.

A seguir, serão apresentados teoremas e definições para o estudo de EDIs com os momentos impulsivos fixados. Consideraremos, portanto, $J_1 = [t_0, \omega)$ e $J_2 = [t_0, \widetilde{\omega})$, com $J_1 \subseteq J_2$.

Definição 1 *Se $x(t) = x(t; t_0, x_0)$ e $y(t) = y(t; t_0, x_0)$ são duas soluções do sistema (6), nos intervalos J_1 e J_2, respectivamente, e $x(t) = y(t), \forall t \in J_1$, então $y(t)$ é dita a continuação de $x(t)$ no intervalo J_2 (continuação à direita).*

Teorema 1 *A solução $x(t) = x(t; t_0, x_0)$ é dita ser continuável no intervalo J_2, se existir a continuação $y(t)$ de $x(t)$ em J_2. Caso contrário, $x(t) = x(t; t_0, x_0)$ é dita ser não continuável e o intervalo J_1 é o máximo intervalo de existência de $x(t)$.*

Definição 2 *A solução $x(t) = x(t; t_0, x_0)$ do sistema (10) é dita ser única quando, dada qualquer outra solução $y(t) = y(t; t_0, y_0)$, $x(t) = y(t)$ dentro do intervalo comum de existência.*

2.1 Exemplos

2.1.1 (LAKSHMIKANTHAM; BAĬNOV; SIMEONOV, 1989 apud BONOTTO, 2005, p. 7)

Dada a condição inicial de $x(0) = 0$, resolver a seguinte EDI.

$$\begin{cases} \dot{x} = 1 + x^2, \ t \neq \frac{k\pi}{4} \\ \Delta x(t) = -1, \ t = \frac{k\pi}{4}, \ k \in \mathbb{Z} \end{cases}.$$

Resolvendo-se apenas a EDO, desconsiderando-se os efeitos impulsivos, tem-se

$$\dot{x} = 1 + x^2 \longrightarrow \frac{dx}{dt} = 1 + x^2.$$

Pela definição de diferencial, obtém-se $dx = \frac{dx}{dt}dt$ e, portanto, $dx = (1 + x^2)dt$.

É feita, então, a separação de variáveis e, em seguida, a integração.

$$\frac{dx}{1 + x^2} = dt$$

o que implica que

$$\int \frac{dx}{1 + x^2} = \int dt.$$

Logo,

$$\tan^{-1}(x) = t + c,$$

em que c é a constante de integração. Portanto, $x(t) = \tan(t + c)$. Aplicando-se a condição inicial fornecida pelo enunciado, obtém-se o valor da constante de integração $c = 0$.

Finalmente, a solução da EDO, desconsiderando os impulsos, será dada por $x(t) = \tan(t)$ e, considerando-se o impulso $\Delta x = -1$ para $t = \frac{k\pi}{4}$, a solução do sistema impulsivo será

$$x(t) = \tan\left(t - \frac{k\pi}{4}\right), \ t \in \left]\frac{k\pi}{4}, \frac{(k+1)\pi}{4}\right].$$

Graficamente, o comportamento do sistema é representado a seguir.

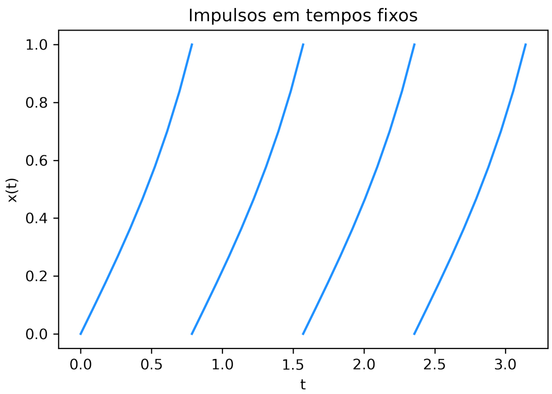

Figura 1: Descontinuidades para $t = \frac{k\pi}{4}$, com $k \in \mathbb{Z}$. (Fonte: Elaborado pelo compilador Python)

2.1.2 (LAKSHMIKANTHAM; BAĬNOV; SIMEONOV, 1989 apud BONOTTO, 2005, p. 9)

Sendo $t \geq 0$ e $k \in \mathbb{Z}_+$, resolver a EDI dada por

$$\begin{cases} \dot{x} = 0, \ t \neq \tau_k(x) \\ \Delta x = x^2 sgn(x) - x, \ t = \tau_k(x). \end{cases}$$

Considere, também, que a superfície $S_k : \ t = \tau_k(x)$ seja descrita por

$$\tau_k(x) = x - 6k, \quad \text{com } |x| < 3.$$

A função $sgn(x)$ é dada por

$$sgn(x) = \begin{cases} 1, \ x > 0 \\ 0, \ x = 0 \\ -1, \ x < 0 \ . \end{cases}$$

Primeiramente, serão calculadas algumas superfícies impulsivas S_k:

$$\begin{cases} k = 0 \longrightarrow S_0 : \tau_0(x) = x \\ k = 1 \longrightarrow S_1 : \tau_1(x) = x - 6 \ . \end{cases}$$

Além disso, como $|x| < 3$, então $-3 < x < 3$.

Considerando-se apenas $t \geq 0$, tem-se

- As soluções $x(t)$ com condição inicial $x(0) = x_0$, tais que $|x_0| \geq 3$ não sofrem impulso, pois não intersectam as superfícies S_k.

- Para as soluções que se iniciam em $(0, x_0)$, com $1 < x_0 < 3$, tais soluções sofrem o efeito impulsivo um número finito de vezes. Por exemplo, a solução $x(t)$, tal que $x(0) = x_0 = \sqrt[4]{2}$, representada no gráfico a seguir, sofre três impulsos.

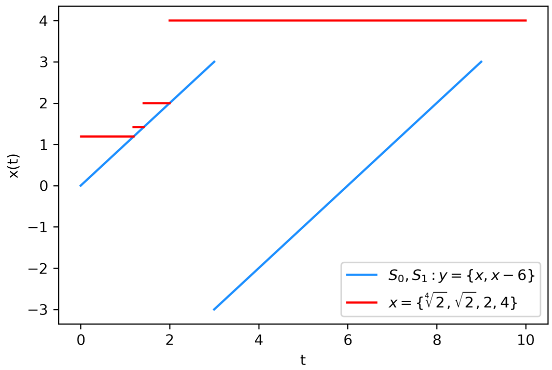

Figura 2: Descontinuidades para tempos variáveis. (Fonte: Elaborado pelo compilador Python)

Na cor azul estão representadas as superfícies impulsivas S_0 e S_1 e, em vermelho, está representado o comportamento da solução $x(t)$, cuja condição inicial é $x_0 = \sqrt[4]{2} \in]1, 3[$.

- Se o ponto inicial $x(0) = x_0$ pertencer ao intervalo $]0, 1[$, a solução $x(t)$ intersectará as superfícies S_k um número infinito de vezes. Por conseguinte, sofrerá infinitos impulsos. O gráfico a seguir ilustra a situação para $x_0 = \frac{1}{2}$.

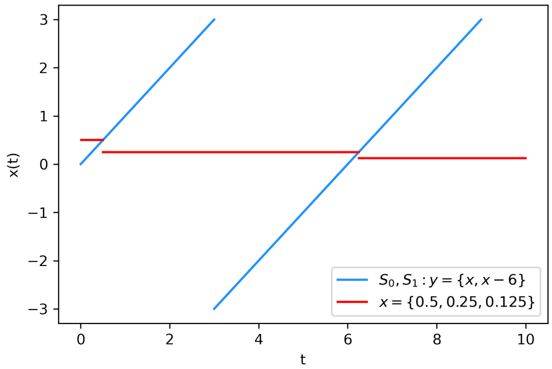

Figura 3: Descontinuidades para tempos variáveis. (Fonte: Elaborado pelo compilador Python)

É notável que $\{x(t_k)\}$ é uma Progressão Geométrica infinita cujo primeiro termo é $x(0) = x_0 = \frac{1}{2}$ e cuja razão é $q = \frac{1}{2}$. Ademais, $\lim_{k\to+\infty} t_k = +\infty$ e $\lim_{k\to+\infty} x(t_k) = 0$.

- Para as soluções em que $-1 < x_0 < 0$, também ocorrerá um número infinito de impulsos. Além disso, $\lim_{k\to+\infty} t_k = 6$ e $\lim_{k\to+\infty} x(t_k) = 0$. Adotemos o caso em que $x_0 = -\frac{1}{2}$. O gráfico a seguir ilustra o comportamento da solução $x(t)$ para essa condição inicial em particular.

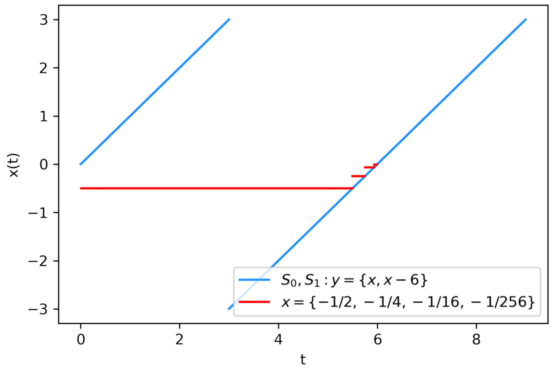

Figura 4: Descontinuidades para tempos variáveis. (Fonte: Elaborado pelo compilador Python)

Ampliando-se o gráfico anterior, visando à obtenção de maiores detalhes do comportamento da solução $x(t)$ em vermelho, tem-se:

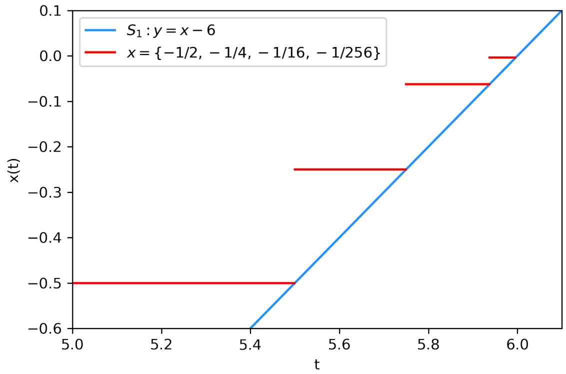

Figura 5: Descontinuidades para tempos variáveis. (Fonte: Elaborado pelo compilador Python)

3 Estabilidade e Fronteira

Dado $k \in \mathbb{Z}^*$, considere o sistema impulsivo dado por

$$\begin{cases} \dot{x}(t) = f(t,x), \ t \neq t_k \\ \Delta x = I_k(x), \ t = t_k \end{cases}.$$

Uma solução será dada por $\psi(t) = \psi(t; t_0, \psi_0)$, com $\psi(t_0^+) = \psi_0 \in \Omega$. Tal solução $\psi(t)$ pode ser classificada, segundo I. Stamova e G. Stamov (2016), em:

a) Estável: se um ponto inicial qualquer $x_0 \in \Omega$ estiver próximo do valor inicial $\psi(t_0^+)$ da solução e, se para todo t posterior ao tempo inicial t_0, $x(t)$ se mantiver próximo da solução $\psi(t)$, então a solução é dita estável. Matematicamente, tem-se:

$$\forall t_0 \in \mathbb{R}, \forall \varepsilon > 0, \exists \delta = \delta(t_0, \varepsilon) > 0$$

$$\left(\forall x_0 \in \Omega : \left\|x_0 - \psi(t_0^+)\right\| < \delta\right) (\forall t \geq t_0) : \|x(t; t_0, x_0) - \psi(t)\| < \varepsilon \ .$$

b) Uniformemente estável: o valor de δ no item $a)$ deve ser independente de $t_0 \in \mathbb{R}$.

c) Atrativa: à medida em que t aumenta, o ponto inicial x_0 tende a se aproximar da solução $\psi(t)$. Matematicamente,

$$(\forall t_0 \in \mathbb{R}), (\exists \lambda = \lambda(t_0) > 0)$$
$$\left(\forall x_0 \in \Omega : \left\|x_0 - \psi(t_0^+)\right\| < \lambda\right), \lim_{t \to \infty} \|x(t; t_0, x_0) - \psi(t)\| < \varepsilon \ .$$

d) Equiatrativa: o ponto inicial x_0 se aproxima da solução $\psi(t)$ somente para valores de $t \geq t_0 + T$, sendo T um valor positivo. Numa linguagem formal, tem-se:

$$(\forall t_0 \in \mathbb{R}), (\exists \lambda = \lambda(t_0) > 0), (\forall \varepsilon > 0), (\exists T = T(T_0, \varepsilon) > 0)$$

$$\left(\forall x_0 \in \Omega : \left\|x_0 - \psi(t_0^+)\right\| < \lambda\right)$$
$$(\forall t \geq t_0 + T) : \|x(t; t_0, x_0) - \psi(t)\| < \varepsilon \ .$$

e) Uniformemente atrativa: os valores de λ e T no item $d)$ devem ser independentes de $t_0 \in \mathbb{R}$.

f) Assintoticamente estável: deve ser estável e atrativa.

g) Uniformemente assintoticamente estável: deve ser uniformemente atrativa e assintoticamente estável.

h) Exponencialmente estável: dado um ponto inicial qualquer x_0, ele se aproxima da solução $\psi(t)$ de maneira exponencial, adicionando-se um fator de correção $\gamma \geq 0$. Matematicamente, tem-se:

$$(\exists \lambda > 0) (\forall \alpha > 0) (\exists \gamma = \gamma(\alpha) > 0) (\forall t_0 \in \mathbb{R})$$
$$\left(\forall x_0 \in \Omega : \left\|x_0 - \psi(t_0^+)\right\| < \alpha\right) (\forall t \geq t_0)$$
$$\|x(t; t_0, x_0) - \psi(t)\| < \gamma(\alpha) \left\|x_0 - \psi(t_0^+)\right\| e^{-\lambda(t - t_0)} \ .$$

4 Funções de Lyapunov

As funções de Lyapunov, também conhecidas como Funções de Energia e usualmente representadas por $V(x)$, lidam ainda com a análise de estabilidade das soluções de equações diferenciais. Pode-se dizer que tais funções tornam o estudo abordado no *Seção 3* deste artigo algo menos teórico e um pouco mais prático e palpável.

Assim, sendo dada a equação $\dot{x} = f(x) \in \mathbb{R}^2$ e sendo x^* um ponto de equilíbrio, Richard Pates (vide item [3] das *Referências*) representa duas possíveis trajetórias.

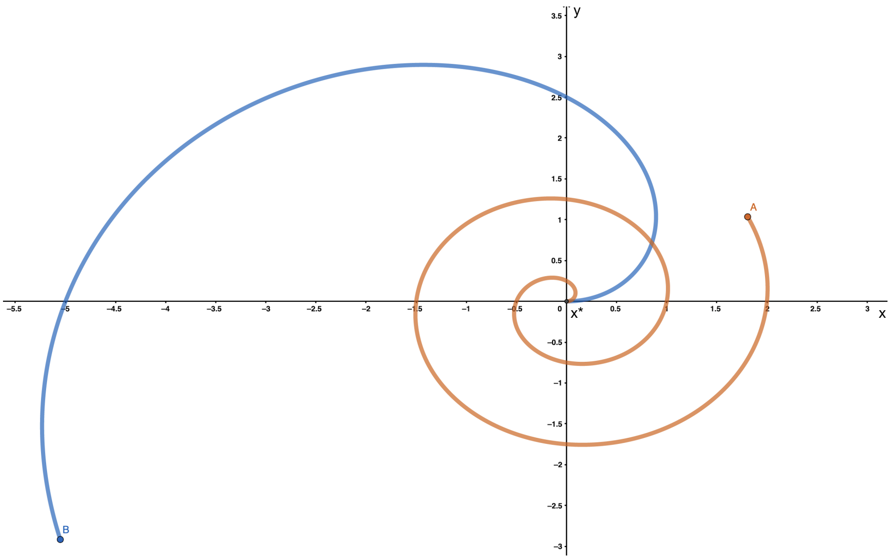

Figura 6: Trajetórias com diferentes condições iniciais A e B convergindo para o ponto comum de estabilidade x^*. (Fonte: Elaborado pelo compilador Geogebra)

A seguir, estão representadas duas superfícies equienergéticas, bem como uma trajetória, solução de $\dot{x} = f(x)$.

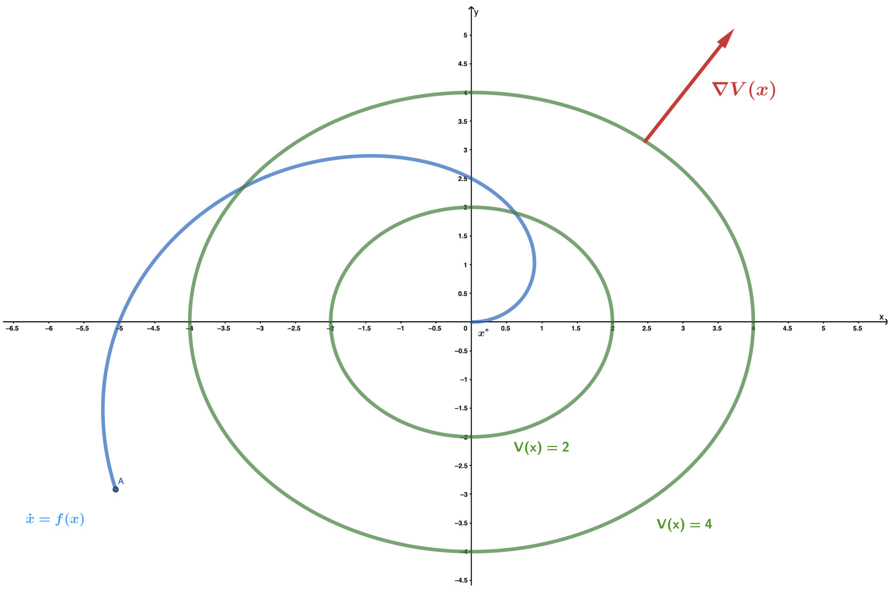

Figura 7: Superfícies equipotenciais de Lyapunov e a trajetória de $\dot{x} = f(x)$. (Fonte: Elaborado pelo compilador Geogebra)

Uma função $V(x)$ é dita ser de Lyapunov, se obedecer às seguintes propriedades:

- $V(x^*) = 0$;
- $V(x) > 0, \forall x \neq x^*$;
- Se $\nabla V(x) \cdot \dot{x} < 0$, então as trajetórias vão de valores superiores a valores inferiores de V.

5 Teorema de Estabilidade de Lyapunov

Pates, em vídeo gravado para a plataforma *YouTube*, vide item [3] das *Referências*, define as Funções de Lyapunov, de acordo com o seguintes tópicos:

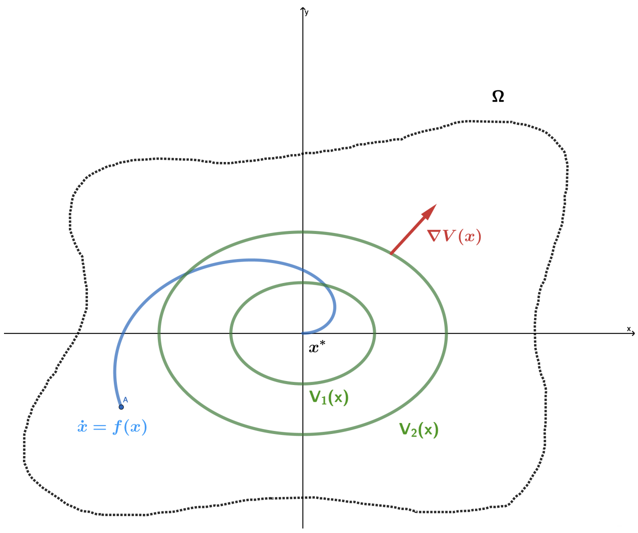

Figura 8: Conjunto aberto $\Omega \subset \mathbb{R}^2$. (Fonte: Elaborado pelo compilador Geogebra)

Seja $\Omega \subset \mathbb{R}^2$, tal que o conjunto das funções de Lyapunov $V(x) \subset \Omega$. Assim, dados $\dot{x} = f(x)$ e $V(x)$ definidos em Ω,

1. $V(x) = 0$, para $x = x^*$;
2. $V(x) > 0$, $\forall x \in \Omega$ e $x \neq x^*$;
3. se $\dot{V}(x) \doteq \nabla V(x) \cdot f(x) \leq 0, \forall x \in \Omega \Rightarrow x^*$ é estável.
   Obs.: se $\dot{V}(x) \leq 0$, então a solução permanece dentro do conjunto aberto Ω.
4. Se $\dot{V}(x) < 0, \forall x \in \Omega - \{x^*\} \Rightarrow x^*$ é localmente e assintoticamente estável.
   Obs: Se $\dot{V}(x) < 0$ (estritamente menor do que zero), então a solução não apenas permanece no espaço Ω, como também converge para o ponto de estabilidade $x^* \in \Omega$.
5. Se $\Omega = \mathbb{R}^n$ ($\mathbb{R}^2$ neste exemplo) e $V(x) \to \infty$ quando

$$\|x\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2} \to \infty,$$

   então x^* é global e assintoticamente estável.

5.1 Exemplo

Provar, utilizando o Teorema de Lyapunov, que o sistema constituído pelo pêndulo simples da imagem a seguir é estável.

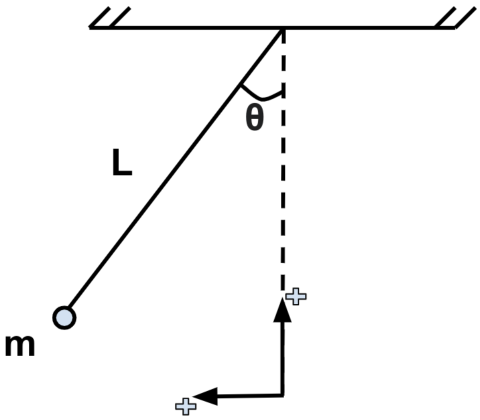

Figura 9: Pêndulo Simples. (Fonte: Elaborado pelo compilador Google Drawings)

Prova: Das relações básicas da Cinemática, tem-se:

$$\begin{cases} v = L\omega = L\dfrac{d\theta}{dt} \\ a = \dfrac{dv}{dt} = L\dfrac{d^2\theta}{dt^2} \end{cases}.$$

Ademais, pela Segunda Lei de Newton, tem-se

$$-mg\sin\theta = ma = mL\frac{d^2\theta}{dt^2}.$$

Multiplicando-se ambos os lados da igualdade por L e remanipulando-se a equação, obtém-se

$$mL^2\ddot{\theta} + mgL\sin\theta = 0.$$

Portanto, esta última equação descreve o comportamento físico do sistema analisado. Ainda desta equação, ao se isolar a aceleração angular do pêndulo, obtém-se $\ddot{\theta} = -\frac{g}{L}\sin\theta$. Definindo-se, portanto, a matriz das variáveis de estado $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$, tem-se

$$\frac{dx}{dt} = \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{d}{dt}\begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{g}{L}\sin\theta \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{g}{L}\sin x_1 \end{bmatrix}.$$

Logo,

$$\underbrace{\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{} = \underbrace{\begin{bmatrix} x_2 \\ -\frac{x}{L}\sin x_1 \end{bmatrix}}_{}$$

ou seja,

$$\dot{x} = f(x) \ .$$

Ainda sobre a definição, o conjunto aberto $\Omega(x_1, x_2)$ é dado por

$$\Omega = \left\{ \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} : -\pi < x_1 < \pi, \|x_2\| < k \right\}, \ com \ k \in \mathbb{R}.$$

Além disso, o ponto $x^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \in \Omega$ é o ponto de equilíbrio do sistema.

Assim, define-se a Função de Lyapunov do sistema como sendo a soma das energias potencial e cinética. Portanto,

$$\begin{aligned} V(\theta) &= mgL(1 - \cos\theta) + \frac{1}{2}mv^2 \\ &= mgL(1 - \cos\theta) + \frac{1}{2}mL^2\omega^2 \\ &= mgL(1 - \cos\theta) + \frac{1}{2}mL^2(\dot{\theta})^2. \end{aligned}$$

No espaço Ω, tem-se

$$V(x) = mgL(1 - \cos x_1) + \frac{1}{2}mL^2x_2^2.$$

Finalmente, para que seja possível concluir sobre a estabilidade do sistema analisado neste exemplo, basta analisar a função $V(x)$ e checar se ela cumpre todos os requisitos do Teorema de Lyapunov.

1. $x^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow V(x^*) = mgL(1 - \cos 0) + \frac{1}{2}mL^2 0^2 = 0 + 0 = 0.$

2. Para todo $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \Omega$, com $x \neq x^*$, $V(x) > 0$.

3. Vale

$$\begin{aligned} \dot{V}(x) = \nabla V(x) \cdot \dot{x} &= \begin{bmatrix} \frac{\partial V}{\partial x_1} \\ \frac{\partial V}{\partial x_2} \end{bmatrix} \cdot \begin{bmatrix} x_2 \\ -\frac{g}{L}\sin x_1 \end{bmatrix} \\ &= \begin{bmatrix} mgL\sin x_1 \\ mL^2x_2 \end{bmatrix} \cdot \begin{bmatrix} x_2 \\ -\frac{g}{L}\sin x_1 \end{bmatrix} \\ &= mgLx_2\sin x_1 - mgLx_2\sin x_1 \\ &= \nabla V(x) \cdot \dot{x} = \nabla V(x) \cdot f(x) = 0 \ . \end{aligned}$$

Como todos os critérios foram obedecidos, então o pêndulo simples constitui um sistema estável.

□

5.2 Aplicação do Teorema de Lyapunov sobre o Exemplo da Subseção 2.4.1

Para a análise de estabilidade da solução do *Exemplo 2.4.1* deste artigo, será aplicado o *Teorema de Lyapunov*.

$$\begin{cases} \dot{x} = 1 + x^2, \ t \neq \frac{k\pi}{4} \\ \Delta x(t) = -1, \ t = \frac{k\pi}{4}, \ k \in \mathbb{Z} \end{cases}$$

A solução já calculada forneceu, desconsiderando-se os efeitos impulsivos,

$$\begin{cases} x(t) = \tan(t) \\ \dot{x}(t) = \sec^2(t), \end{cases}$$

e o comportamento gráfico da solução da EDI foi mostrado na Figura 1 deste relatório.

Assim, assumindo-se que $x(t)$ seja a posição de uma partícula de massa m, num tempo t, pode-se, portanto, definir uma função de energia $V(x)$ associada. Assim:

$$V(x) = K(x) + U(x),$$

em que $K(x)$ é a energia cinética da partícula e $U(x)$ é a sua energia potencial. Portanto, tem-se

$$V(x) = \frac{1}{2}mv^2 + mgx,$$

em que g é a norma da aceleração do campo gravitacional local. Prosseguindo-se no desenvolvimento da expressão, tem-se

$$V(x) = \frac{1}{2}m(\dot{x})^2 + mgx \quad .$$

Dado que x e $\dot{x}$ são ambas funções do tempo, então, para $0 \leq t < \frac{\pi}{4}$, tem-se

$$\begin{aligned} V(t) &= \frac{1}{2}m(\sec^2(t))^2 + mg\tan(t) \\ &= \frac{1}{2}m\sec^4(t) + mg\tan(t). \end{aligned}$$

Obtendo-se a derivada temporal do potencial, tem-se

$$\dot{V}(t) = 2m\sec^4(t)\tan(t) + mg\sec^2(t), \quad 0 \leq t < \frac{\pi}{4}.$$

Analisando-se as condições do *Teorema de Lyapunov*, tem-se

1. $\nexists\, t^* \in \left[0, \frac{\pi}{4}\right)$, tal que $V(t^*) = 0$;
2. $\forall\, t \in \left[0, \frac{\pi}{4}\right)$, $V(t) > 0$;
3. $\forall\, t \in \left[0, \frac{\pi}{4}\right)$, $\dot{V}(t).\dot{x}(t) > 0$.

Pelos itens 1 e 3, o *Teorema de Lyapunov* garante que a solução $x(t) = \tan(t)$, na ausência de impulsos, é instável. O que garante, portanto, a estabilidade da solução da EDI é justamente a presença do efeito impulsivo

$$\Delta x(t) = -1, \quad \text{para } t = \frac{k\pi}{4}, \quad \text{com } k \in \mathbb{Z}.$$

6 Métodos Numéricos para EDOs - Abordagem Analítica

Este tópico trará uma abordagem analítica de métodos numéricos para a resolução de EDOs, como o *Método de Euler*, os *Métodos de Série de Taylor* e os *Métodos de Runge-Kutta*. Esta parte se baseou nos trabalhos de Ruggiero e Lopes (1996), além do trabalho desenvolvido por Azevedo (2021).

6.1 Método de Euler

Dado o PVI

$$\begin{cases} y' = f(x, y) \\ y(x_0) = y_0, \end{cases} \tag{7}$$

o passo simples do método é representado por h e é definido como sendo

$$h = x_{i+1} - x_i, \ \forall i \in \mathbb{N}.$$

Assim, dado que $y(x_0) = y_0$ é conhecido, bem como o valor da derivada $y'(x)$ na abscissa $x = x_0$, obtém-se a reta $r_0(x)$ tangente ao gráfico de $y(x)$ no ponto (x_0, y_0). Ou seja,

$$\begin{aligned} r_0(x) &= y'(x_0) \times (x - x_0) + y(x_0) \\ &= f(x_0, y_0) \times (x - x_0) + y_0. \end{aligned}$$

A partir de tal reta, pode-se obter uma aproximação para o valor y_1 da função avaliada na abscissa do passo seguinte $x_1 = x_0 + h$. Assim,

$$\begin{aligned} y_1 = y(x_1) \approx r_0(x_1) &= y'(x_0) \times (x_1 - x_0) + y(x_0) \\ &= f(x_0, y_0) \times (x_0 + h - x_0) + y_0 \end{aligned}$$

e, portanto,

$$y_1 = y(x_1) \approx h \times f(x_0, y_0) + y_0.$$

O raciocínio é então repetido com o par ordenado (x_1, y_1) para o cálculo de $y_2 = y(x_2)$ e assim sucessivamente. De maneira genérica, o *Método de Euler* fornece

$$y_{k+1} = y_k + h \times f(x_k, y_k), \ \forall k \in \mathbb{N},$$

sendo $h = x_{k+1} - x_k$ o *passo* adotado.

Graficamente, tem-se:

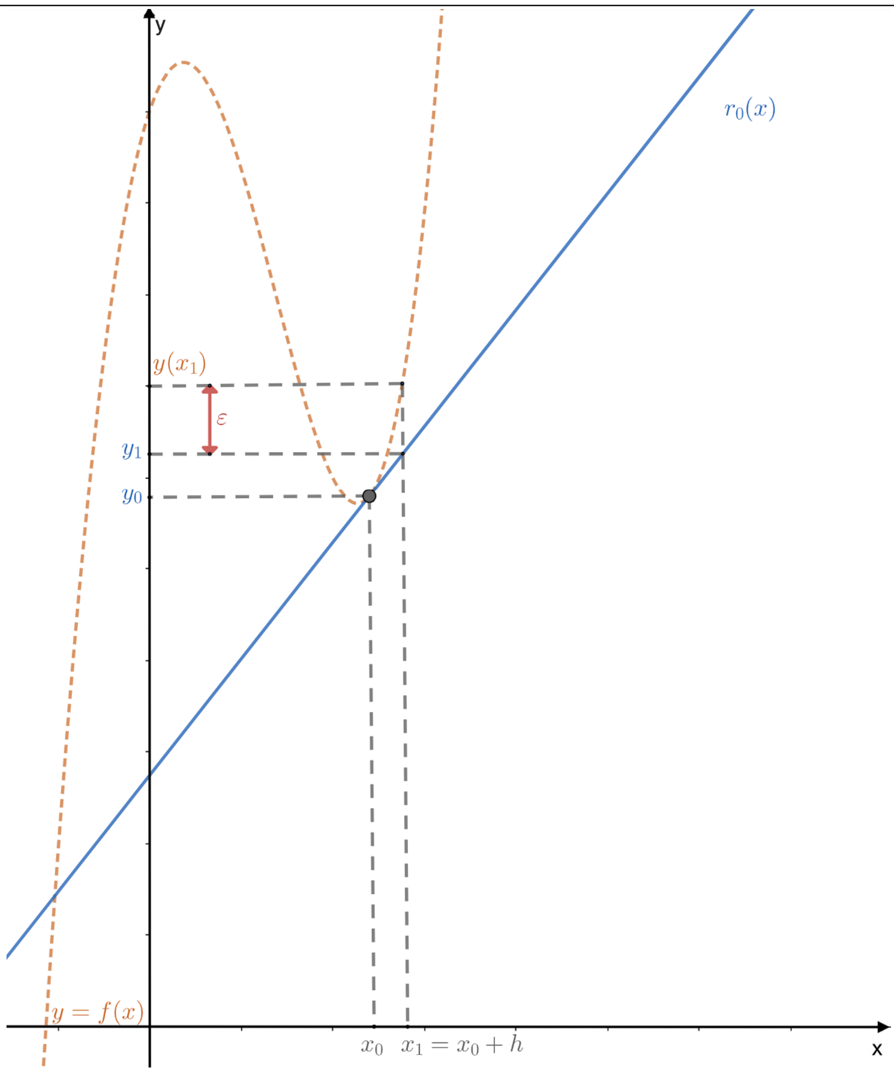

Figura 10: Representação gráfica do *Método de Euler*. (Fonte: Elaborado pelo compilador Geogebra)

6.1.1 Exemplo

Dado o PVI a seguir, utilizar o *Método de Euler* para aproximar $y(0.04)$ com erro inferior a 5×10^{-4}.

$$\begin{cases} y' = y \\ y(0) = 1 \end{cases}.$$

Para o *Método de Euler*, o erro é dado por

$$e(x_n) = \frac{h^2}{2!} \times y(\zeta_{x_n})''.$$

Neste caso, a solução analítica é conhecida e vale $y(x) = e^x$. Portanto, tem-se que

$$\begin{aligned} M_2 &= \max\{y''(x_n) \mid 0 \le x \le 0.04\} = \max\{e^x \mid 0 \le x \le 0.04\} \\ &= e^{0.04} = 1.0408. \end{aligned}$$

Dado que $e(x_n) \leq 5 \times 10^{-4}$, então

$$\frac{1.0408}{2} \times h^2 \leq 5 \times 10^{-4} \to h \leq 0.0310.$$

Agora, basta escolher o maior valor de h a fim de se trabalhar com pontos igualmente espaçados. Portanto, será escolhido $h = 0.02$, pois deseja-se calcular $y(0.04)$. Assim, tem-se $x_0 = 0$, $x_1 = h = 0.02$ e $x_2 = 2h = 0.04$.

Ademais,

$$\begin{aligned} y_1 &= y_0 + h \times y'(0) \to y_1 = 1 + 0.02 \times 1 = 1.02 \\ y_2 &= y_1 + h \times y'(1) \to y_2 = 1.02 + 0.02 \times 1.02 = 1.0404 \end{aligned}.$$

Logo, o *Método de Euler* forneceu

$$y_2 = y(x_2) = y(0.04) = 1.0404.$$

Finalmente, é possível se fazer a verificação do erro ε:

$$\varepsilon = |e^{0.04} - 1.0404| = 4 \times 10^{-4} < 5 \times 10^{-4}.$$

Portanto, o erro cometido pelo método foi inferior ao erro máximo permitido.

6.2 Métodos de Série de Taylor

Considerando-se o PVI de (7), a aproximação da função $y(x)$ em torno da abscissa $x = x_n$ é dada por

$$\begin{aligned} y(x) = y(x_n) + (x - x_n) \times y'(x_n) + \frac{(x - x_n)^2}{2!} \times y''(x_n) + \\ + \frac{(x - x_n)^3}{3!} \times y'''(x_n) + \cdots + \frac{(x - x_n)^k}{k!} \times y^{(k)}(x_n) + e(x), \end{aligned}$$

em que $e(x)$ é o erro de truncamento, descrito por

$$e(x) = \frac{(x - x_n)^{k+1}}{(k+1)!} \times y^{(k+1)}(\zeta_x),\ \ \zeta_x \in]x, x_n[.$$

Com isso, faz-se um raciocínio análogo ao *Método de Euler*, visando à obtenção da aproximação numérica de $y_{k+1} = y(x_{k+1})$, partir dos valores de x_k e y_k. Ou seja, y_{k+1} será dado por

$$\begin{aligned} y_{k+1} = y(x_{k+1}) = y(x_k) + (x_{k+1} - x_k) \times y'(x_k) + \frac{(x_{k+1} - x_k)^2}{2!} \times y''(x_k) + \\ + \frac{(x_{k+1} - x_k)^3}{3!} \times y'''(x_k) + \cdots + \frac{(x_{k+1} - x_k)^n}{n!} \times y^{(n)}(x_k). \end{aligned}$$

Assim, o *Polinômio de Taylor* de ordem n será

$$y_{k+1} = y_k + h \times {y_k}' + \frac{h^2}{2!} \times {y_k}'' + \frac{h^3}{3!} \times {y_k}''' + \cdots + \frac{h^n}{n!} \times y_k^{(n)},$$

cujo erro é descrito por

$$|e(x_{n+1})| \leq M_{n+1} \times \frac{h^{n+1}}{(n+1)!}.$$

Tendo-se em mente o objetivo de se resolver o PVI (7), adotando-se o *Polinômio de Taylor de Ordem 2*, tem-se que

$$y_{k+1} = y_k + h \times {y_k}' + \frac{h^2}{2!} \times {y_k}'',$$

sendo que

- vale ${y_k}' = y'(x_k) = f(x_k, y_k)$ e
- valem as igualdades

$$\begin{aligned} {y_k}'' = y''(x_k) = f'(x_k, y_k) &= \frac{d}{dx} f'\left(x_k, y(x_k)\right) \\ &= \frac{\partial}{\partial x} f(x_k, y(x_k)) + \frac{\partial}{\partial y} f(x_k, y(x_k)) \times \frac{\partial}{\partial x} y(x_k) \end{aligned}$$

  e

$$\boxed{{y_k}'' = y''(x_k) = f_x(x_k, y_k) + f_y(x_k, y_k) \times f(x_k, y_k)} \ .$$

6.2.1 Exemplo

Dado o PVI

$$\begin{cases} xy' = x - y \\ y(2) = 2, \end{cases}$$

calcular $y(2.1)$ utilizando a Série de Taylor de Ordem 2.

O Polinômio de Taylor será construído em torno do ponto de abscissa $x = 2$. Assim,

$$xy' = x - y \leftrightarrow y'(x) = 1 - \frac{y(x)}{x}.$$

Para $x = 2$, tem-se

$$y'(2) = 1 - \frac{y(2)}{2} = 1 - \frac{2}{2} = 1 - 1 \to y'(2) = 0.$$

Para a obtenção da derivada de segunda ordem no ponto de abscissa $x = 2$, basta derivar a EDO do problema em relação a x, ou seja,

$$y' + xy'' = 1 - y' \leftrightarrow xy'' = 1 - 2y' \leftrightarrow y''(x) = \frac{1}{x} - \frac{2y'(x)}{x}.$$

Para $x = 2$, tem-se $y''(2) = \frac{1}{2}$.

Finalmente, o Polinômio de Taylor será dado por

$$\begin{aligned} y(x) &= y(2) + (x-2) \times y'(2) + \frac{(x-2)^2}{2} \times y''(2) \\ &= 2 + \frac{1}{4}(x-2)^2 + \frac{1}{6}(x-2)^3. \end{aligned}$$

Substituindo-se, portanto, $x = 2.1$, o valor da ordenada será de

$$\begin{aligned} y(2.1) &= 2 + \frac{1}{4} \times (0.1)^2 + \frac{1}{6} \times (0.1)^3 \\ &= 2 + 0.25 \times 0.01 = 2.00238. \end{aligned}$$

6.3 Métodos de Runge-Kutta

São uma otimização dos Métodos da Série de Taylor e possuem as seguintes propriedades:

- São de passo unitário;
- Não exigem o cálculo de qualquer derivada de $f(s, y)$, no entanto, é necessário calcular $f(x, y)$ em vários pontos;
- Após se realizar a expansão de $f(x, y)$ por Taylor em torno de (x_n, y_n) e agrupar os termos semelhantes, sua expressão coincide com a do Método da Série de Taylor de mesma ordem.

6.3.1 Método de Runge-Kutta de Primeira Ordem

Coincide com o *Método de Euler*, a menos pelo cálculo da derivada no ponto. Em outras palavras, por *Euler*,

$$y_{n+1} = y_n + h \times y'_n, \; n \in \mathbb{N}.$$

Substituindo-se o cálculo da derivada pelo cálculo da função de duas variáveis $y'_n = f(x_n, y_n)$, o *Método de Runge-Kutta de Primeira Ordem* será dado por

$$y_{n+1} = y_n + h \times f(x_n, y_n), \; n \in \mathbb{N}.$$

6.3.2 Método de Runge-Kutta de Segunda Ordem

Para este método, basta fazer a expansão do Polinômio de Taylor de Segunda Ordem, substituir os cálculos das derivadas e agrupar os termos semelhantes. Assim, tal método fornecerá

$$y_{n+1} = y_n + h \times (1-w) \times f(x_n, y_n) + h \times w \times f\left(x_n + \frac{h}{2w}, y_n + \frac{h}{2w} f(x_n, y_n)\right),$$

com $n \in \mathbb{N}$ e $w \neq 0$.

6.3.3 Método de Runge-Kutta de Terceira Ordem

$$y_{n+1} = y_n + \frac{2}{9} \times k_1 + \frac{1}{3} \times k_2 + \frac{4}{9} \times k_3, \ n \in \mathbb{N},$$

sendo que

$$\begin{aligned} k_1 &= h \times f(x_n, y_n), \\ k_2 &= h \times f\left(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right), \\ k_3 &= h \times f\left(x_n + \frac{3h}{4}, y_n + \frac{3k_2}{4}\right). \end{aligned}$$

6.3.4 Método de Runge-Kutta de Quarta Ordem

$$y_{n+1} = y_n + \frac{1}{6} \times (k_1 + 2k_2 + 2k_3 + k_4), \ n \in \mathbb{N},$$

em que

$$\begin{aligned} k_1 &= h \times f(x_n, y_n), \\ k_2 &= h \times f\left(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right), \\ k_3 &= h \times f\left(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}\right), \\ k_4 &= h \times f\left(x_n + h, y_n + k_3\right). \end{aligned}$$

7 Métodos Numéricos para EDOs - Abordagem Computacional

Nesta seção, serão implementados, em linguagem *Python*, os *Métodos de Runge-Kutta* para a resolução de EDOs. Para isso, a lógica de todos os métodos se resume a iniciar uma lista de valores para x e para y, adicionando-se, inicialmente o ponto de partida (x_0, y_0). Em seguida, calcula-se os valores de x_1 e y_1, adicionando-os ao final das respectivas listas.

A partir do valor do passo h adotado, calcula-se o número n de iterações do método. Durante as iterações, os valores de x_i e y_i calculados são adicionados às listas. Terminadas as iterações os últimos valores das listas correspondem à resposta final do problema.

7.1 Runge-Kutta de Primeira Ordem - Implementação

```python
def runge_kutta_ordem_1(x_f, x_0, y_0, h, f):

  # Inicia as listas de sequencias de valores para x e para y
  x_values = []
  y_values = []

  # Adiciona os valores iniciais x_0 e y_0 nas respectivas listas
```

```python
    x_values.append(x_0)
    y_values.append(y_0)

    # Calcula os proximos valores x_1 e y_1
    x_1 = x_0 + h
    y_1 = y_0 + h * f(x_0, y_0)

    # Adiciona os valores calculados x_1 e y_1 nas respectivas listas
    x_values.append(x_1)
    y_values.append(y_1)

    # Numero de iteracoes para calcular y_f = y(x_f), partindo-se do ponto (x_0,
     y_0)
    n = int( (x_f - x_0) / h )
    for i in range(1, n):

      # Atualiza os valores de x_0 e y_0
      x_0 = x_1
      y_0 = y_1

      # Recalcula os valores de x_1 e y_1
      x_1 = x_0 + h
      y_1 = y_0 + h * f(x_0, y_0)

      # Adiciona os valores calculados x_1 e y_1 nas respectivas listas
      x_values.append(x_1)
      y_values.append(y_1)

    return x_values, y_values
```

Código 1: Código do Método de Runge-Kutta de Primeira Ordem

7.2 Runge-Kutta de Segunda Ordem - Implementação

```python
def runge_kutta_ordem_2(x_f, x_0, y_0, h, f):

    # Inicia as listas de sequencias de valores para x e para y
    x_values = []
    y_values = []

    # Adiciona os valores iniciais x_0 e y_0 nas respectivas listas
    x_values.append(x_0)
    y_values.append(y_0)

    # Calcula os proximos valores x_1 e y_1
    x_1 = x_0 + h
    y_1 = y_0 + (h/2) * ( f(x_0, y_0) + f(x_0 + h, y_0 + h * f(x_0, y_0)) )

    # Adiciona os valores calculados x_1 e y_1 nas respectivas listas
    x_values.append(x_1)
    y_values.append(y_1)

```

```
  # Numero de iteracoes para calcular y_f = y(x_f), partindo-se do ponto (x_0, y_0)
  n = int( (x_f - x_0) / h)

  for i in range(1, n):

    # Atualiza os valores de x_0 e y_0
    x_0 = x_1
    y_0 = y_1

    # Recalcula os valores de x_1 e y_1
    x_1 = x_0 + h
    y_1 = y_0 + (h/2) * ( f(x_0, y_0) + f(x_0 + h, y_0 + h * f(x_0, y_0)) )

    # Adiciona os valores calculados x_1 e y_1 nas respectivas listas
    x_values.append(x_1)
    y_values.append(y_1)

  return x_values, y_values
```

Código 2: Código do Método de Runge-Kutta de Segunda Ordem

7.3 Runge-Kutta de Terceira Ordem - Implementação

```
def runge_kutta_ordem_3(x_f, x_0, y_0, h, f):

  # Inicia as listas de sequencias de valores para x e para y
  x_values = []
  y_values = []

  # Adiciona os valores iniciais x_0 e y_0 nas respectivas listas
  x_values.append(x_0)
  y_values.append(y_0)

  # Calculo de k_1, k_2 e k_3
  k_1 = h * f(x_0, y_0)
  k_2 = h * f(x_0 + h / 2, y_0 + k_1 / 2)
  k_3 = h * f(x_0 + 3 * h / 4, y_0 + 3 * k_2 / 4)

  # Calcula os proximos valores x_1 e y_1
  x_1 = x_0 + h
  y_1 = y_0 + 2 / 9 * k_1 + 1 / 3 * k_2 + 4 / 9 * k_3

  # Adiciona os valores calculados x_1 e y_1 nas respectivas listas
  x_values.append(x_1)
  y_values.append(y_1)

  # Numero de iteracoes para calcular y_f = y(x_f), partindo-se do ponto (x_0, y_0)
  n = int( (x_f - x_0) / h )

  for i in range(1, n):
```

```

        # Atualiza os valores de x_0 e y_0
        x_0 = x_1
        y_0 = y_1

        # Atualiza os valores de k_1, k_2 e k_3
        k_1 = h * f(x_0, y_0)
        k_2 = h * f(x_0 + h / 2, y_0 + k_1 / 2)
        k_3 = h * f(x_0 + 3 * h / 4, y_0 + 3 * k_2 / 4)

        # Recalcula os valores de x_1 e y_1
        x_1 = x_0 + h
        y_1 = y_0 + 2 / 9 * k_1 + 1 / 3 * k_2 + 4 / 9 * k_3

        # Adiciona os valores calculados x_1 e y_1 nas respectivas listas
        x_values.append(x_1)
        y_values.append(y_1)

    return x_values, y_values
```

Código 3: Código do Método de Runge-Kutta de Terceira Ordem

7.4 Exemplo numérico

Dado o PVI a seguir, calcular $y(1)$ pelos Métodos de Runge-Kutta de primeira, segunda e terceira ordens. Adotar como passos $h = 1$, $h = 0.5$ e $h = 0.1$. Em seguida, comparar com a solução exata do problema, dada por

$$y(x) = 1000 \times e^{0.04x} \quad \text{e} \quad y(1) = 1040.8108.$$

Então,

$$\begin{cases} y' = 0.04 \times y \\ y(0) = 1000 \end{cases}.$$

Ao se aplicar os algoritmos, as tabelas a seguir resumem o desempenho dos três métodos de Runge-Kutta para cada valor do passo h adotado.

Tabela 1: Métodos de Runge-Kutta com passo $h = 1$

Ordem de Runge	$y(1)$ calculado	Erro
1	1040.00000	0.81077
2	1040.80000	0.01077
3	1040.81067	0.00011

Tabela 2: Métodos de Runge-Kutta com passo $h = 0.5$

Ordem de Runge	$y(1)$ calculado	Erro
1	1040.40000	0.41077
2	1040.80804	0.00273
3	1040.81076	1.36573e-05

Tabela 3: Métodos de Runge-Kutta com passo $h = 0.1$

Ordem de Runge	$y(1)$ calculado	Erro
1	1040.72773	0.08304
2	1040.81066	0.00011
3	1040.81077	1.10665e-07

A partir de tais tabelas, conclui-se que quanto menor a magnitude do passo h adotado e quanto maior for a ordem do *Método de Runge-Kutta*, menor será o erro cometido e, por conseguinte, maior será a precisão da resposta.

Para ilustrar isso, as figuras a seguir representam as iterações de cada método, variando-se os valores do passo h adotado.

7.4.1 Passo $h = 1$

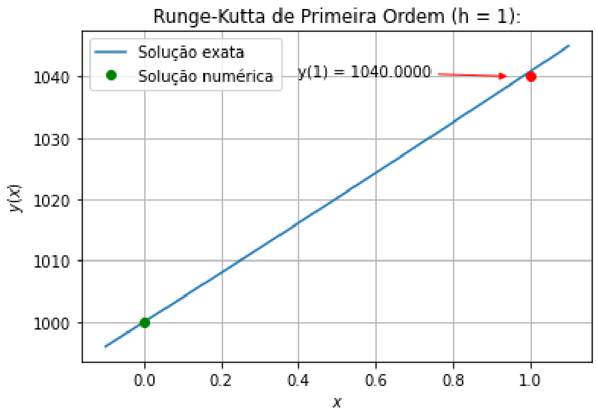

Figura 11: *Runge-Kutta* de ordem 1 e passo $h = 1$

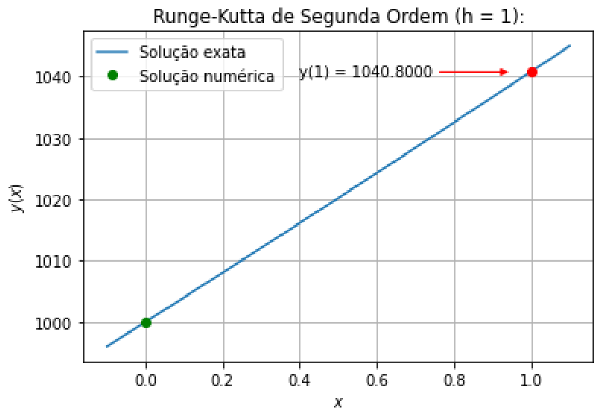

Figura 12: *Runge-Kutta* de ordem 2 e passo $h = 1$

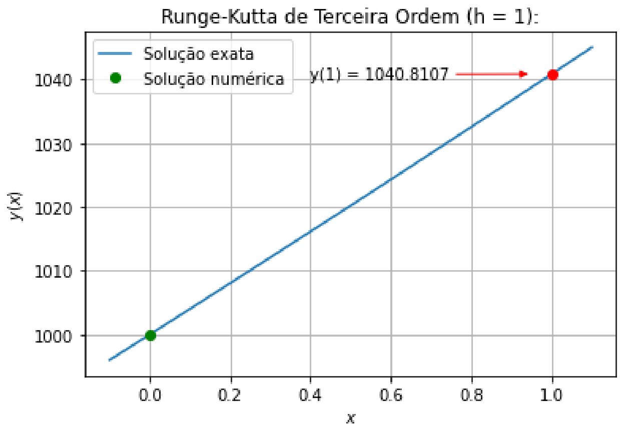

Figura 13: *Runge-Kutta* de ordem 3 e passo $h = 1$

7.4.2 Passo $h = 0.5$

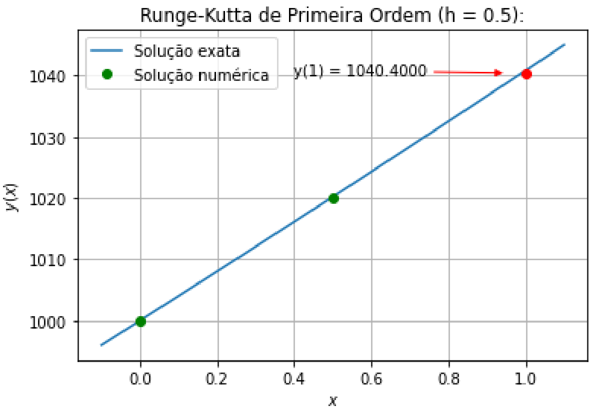

Figura 14: *Runge-Kutta* de ordem 1 e] passo $h = 0.5$

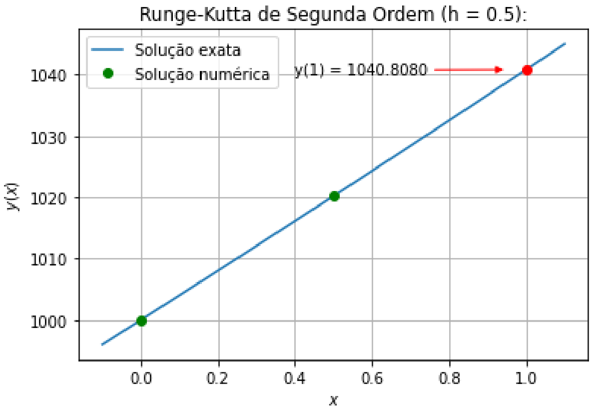

Figura 15: *Runge-Kutta* de ordem 2 e passo $h = 0.5$

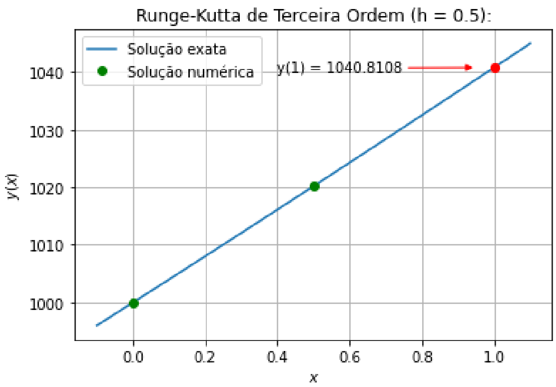

Figura 16: *Runge-Kutta* de ordem 3 e passo $h = 0.5$

7.4.3 Passo $h = 0.1$

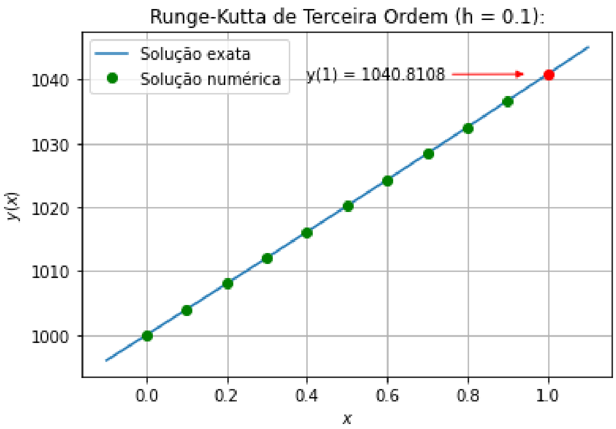

Figura 19: *Runge-Kutta* de ordem 3 e passo $h = 0.1$

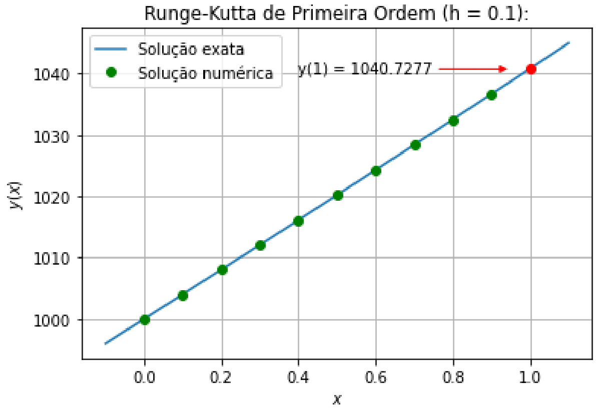

Figura 17: *Runge-Kutta* de ordem 1 e passo $h = 0.1$

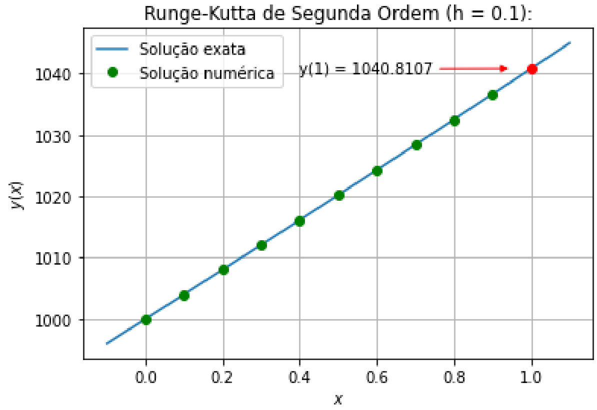

Figura 18: *Runge-Kutta* de ordem 2 e passo $h = 0.1$

8 Método Numérico para a resolução de Equações Diferenciais Impulsivas (EDIs)

Nesta seção, será proposto um método numérico para a resolução de EDIs. Trata-se de uma adaptação dos Métodos de Runge-Kutta. Tal método será apresentado através de um exemplo. Sendo assim, seja dado o seguinte PVI com impulso

$$\begin{cases} y' = 1 + y^2, \ x \neq \frac{k\pi}{4}, \ k \in \mathbb{Z} \\ \Delta y(x) = -1, \ x = \frac{k\pi}{4} \\ y(0) = 0. \end{cases}$$

A resolução deste problema começa pela definição de algumas funções auxiliares, como a função da equação diferencial $f(x, y) = 1 + y^2$, além da função impulsiva $\Delta y(x) = -1$.

```
import numpy as np
import matplotlib.pyplot as plt

# Funcao da EDO
def f(x, y):
    return 1 + y ** 2

# Funcao Impulsiva
def delta_y():
    return -1
```

Código 4: Funções auxiliares

Ademais, foram definidas outras duas funções auxiliares para verificar se o atual valor da abscissa x está sobre uma superfície impulsiva e outra para fornecer o valor da abscissa referente à próxima superfície impulsiva.

```python
# Retorna 'True' se o atual valor da abscissa estiver sobre uma superficie
  impulsiva
# Retorna 'False', caso contrario
def is_on_impulsive_surface(current_x, next_impulsive_x, step):
  if np.abs(current_x - next_impulsive_x) < step:
    return True
  return False

# Retorna o valor da abscissa referente a proxima superficie impulsiva
def get_next_impulsive_surface(current_x):
  k = 0
  while True:
    if current_x > (k * np.pi) / 4:
      k += 1
    else:
      return (k * np.pi) / 4
```

Código 5: Outras funções auxiliares

O último método auxiliar utilizado foi feito para se ter o *plot* da função exata, solução do problema. Ou seja, foi feito para se obter o gráfico de

$$y(x) = \tan\left(x - \frac{k\pi}{4}\right),\ k \in \mathbb{Z}.$$

```python
# Solucao exata do problema: y(x) = tan(x - k pi/4)
def funcao_exata():
  x_values = list(np.arange(0, 2 * np.pi, 0.001))
  y_values = []

  k = 0
  for x in x_values:
    y = np.tan(x - k * np.pi/4)
    if y >= 1:
      k = k + 1
      y = 1

    y_values.append(y)
  return x_values, y_values
```

Código 6: Solução exata do PVI impulsivo

A partir de tais métodos auxiliares, pôde-se adaptar o *Método de Runge-Kutta*, neste caso, de ordem 3, a fim se resolver o PVI com a presença de impulsos em múltiplos inteiros de $\pi/4$.

```python
def runge_kutta_ordem_3_impulsivo(x_f, x_0, y_0, h, f):
  # Inicia as listas de sequencias de valores para x e para y
  x_values = []
  y_values = []

  # Adiciona os valores iniciais x_0 e y_0 nas respectivas listas
  x_values.append(x_0)
  y_values.append(y_0)
```

```
9
10    # Calculo de k_1, k_2 e k_3
11    k_1 = h * f(x_0, y_0)
12    k_2 = h * f(x_0 + h / 2, y_0 + k_1 / 2)
13    k_3 = h * f(x_0 + 3 * h / 4, y_0 + 3 * k_2 / 4)
14
15    # Calcula os proximos valores x_1 e y_1
16    x_1 = x_0 + h
17    y_1 = y_0 + 2 / 9 * k_1 + 1 / 3 * k_2 + 4 / 9 * k_3
18
19    # Adiciona os valores calculados x_1 e y_1 nas respectivas listas
20    x_values.append(x_1)
21    y_values.append(y_1)
22
23    # Numero de iteracoes para calcular y_f = y(x_f), partindo-se do ponto (x_0,
      y_0)
24    n = int( (x_f - x_0) / h )
25
26    for i in range(1, n):
27
28      # Obtem o valor da abscissa onde ocorrera o proximo impulso
29      next_impulsive_x = get_next_impulsive_surface(x_1)
30      if is_on_impulsive_surface(x_1, next_impulsive_x, h) == True:
31        y_1 = y_1 + delta_y()
32
33      # Atualiza os valores de x_0 e y_0
34      x_0 = x_1
35      y_0 = y_1
36
37      # Atualiza os valores de k_1, k_2 e k_3
38      k_1 = h * f(x_0, y_0)
39      k_2 = h * f(x_0 + h / 2, y_0 + k_1 / 2)
40      k_3 = h * f(x_0 + 3 * h / 4, y_0 + 3 * k_2 / 4)
41
42      # Recalcula os valores de x_1 e y_1
43      x_1 = x_0 + h
44      y_1 = y_0 + 2 / 9 * k_1 + 1 / 3 * k_2 + 4 / 9 * k_3
45
46      # Adiciona os valores calculados x_1 e y_1 nas respectivas listas
47      x_values.append(x_1)
48      y_values.append(y_1)
49
50    return x_values, y_values
```

Código 7: Código do Método de Runge-Kutta de Terceira Ordem Impulsivo

Observar que a adaptação do *Método de Runge-Kutta* tradicional se encontra entre as linhas 29 e 32. Nelas, faz-se a checagem se o ponto analisado se encontra numa região impulsiva. Em caso afirmativo, aplica-se o impulso antes da continuação do laço de repetição.

Ao se executar o código, percebe-se que, quanto maior a quantidade de impulsos pelos quais o método passa, maior será o erro entre a solução numérica e a solução analítica exata do problema. Assim, cabe ao programador ajustar a dimensão do passo h para que a solução desejada fique dentro da faixa de erro permitida.

O comportamento da solução numérica fica claro quando se analisa os gráficos. O código a seguir foi utilizado para o *plot* da solução numérica com passo $h = 0.1$, no intervalo de $x = 0$ até $x = 2\pi$.

```
# Definicao dos parametros
x_f = 6.28
x_0 = 0.0
y_0 = 0.0
h = 0.1

# Solucao analitica exata
x_exata, y_exata = funcao_exata()
plt.plot(x_exata, y_exata, label='Solucao exata')

# Solucao numerica (Runge-Kutta de terceira ordem impulsiva)
x_values, y_values = runge_kutta_ordem_3_impulsivo(x_f, x_0, y_0, h, f)
plt.plot(x_values, y_values, 'go', label='Solucao numerica')

plt.xlabel('$x$')
plt.ylabel('$y(x)$')
plt.legend()
plt.grid()
plt.title(f'Runge-Kutta Impulsiva de Terceira Ordem (h = {h}):')
plt.show()
```

Código 8: Código para o *plot* das soluções exata e numérica

A execução deste último código forneceu o seguinte gráfico:

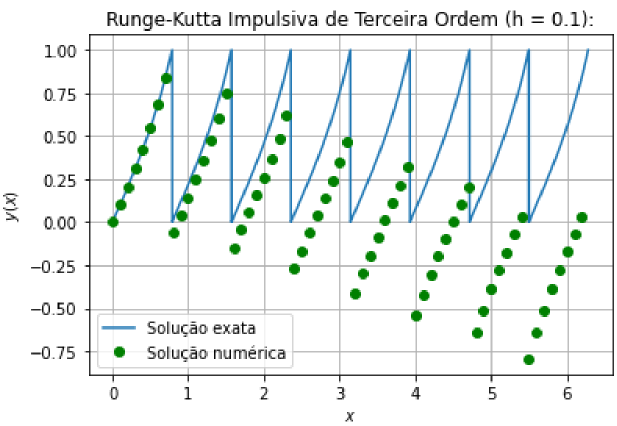

Figura 20: *Runge-Kutta* impulsivo de ordem 3 e passo $h = 0.1$

Ao se analisar o gráfico anterior, nota-se que, a partir do segundo efeito impulsivo, em $x = \pi/2$, a solução numérica começa a divergir significativamente da solução exata.

Uma maneira de se contornar esse fenômeno é diminuir o passo h. Assim, para $h = 0.01$, tem-se:

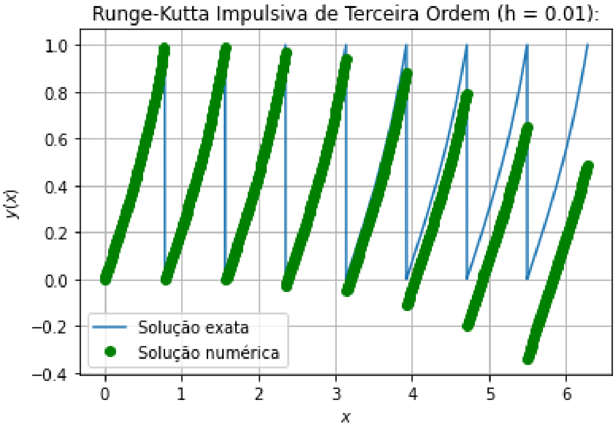

Figura 21: *Runge-Kutta* impulsivo de ordem 3 e passo $h = 0.01$

Para $h = 0.01$, o erro do método numérico começou a ser significativo a partir apenas do sexto impulso. Isso mostra, portanto, uma melhora significativa com relação ao método anterior de passo $h = 0.1$. Finalmente, para $h = 0.001$, o erro praticamente se anula no intervalo de $x = 0$ a $x = 2\pi$, como mostra a figura a seguir.

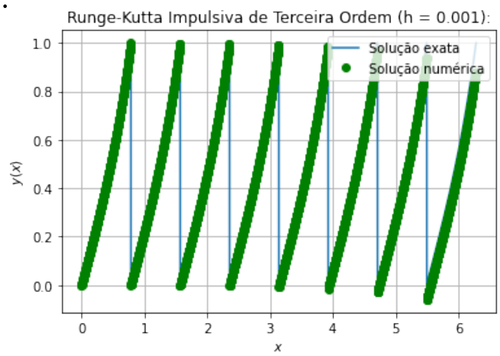

Figura 22: *Runge-Kutta* impulsivo de ordem 3 e passo $h = 0.001$

9 Conclusão

Este artigo apresentou alguns exemplos da vasta aplicabilidade das Equações Diferenciais Impulsivas, além de abordar a teoria básica do assunto. Analisou também a estabilidade de sistemas impulsivos, através do *Teorema de Lyapunov*, concluindo-se, assim, que a ausência de impulsos em alguns sistemas pode torná-los instáveis. Logo, uma maneira de estabilizar tais sistemas, originalmente instáveis, é através da aplicação de efeitos impulsivos periódicos.

Já a segunda parte apresentou alguns dos principais métodos de passo constante para a resolução numérica de Equações Diferenciais Ordinárias, com abordagem através de exemplos. Propôs também uma implementação em linguagem *Python* dos *Métodos de Runge-Kutta* e estendeu tais implementações para sistemas impulsivos. Conclui-se, assim, que apesar de a presença de impulsos ser um fator que pode estabilizar sistemas, também pode ser um fator responsável pela divergência das soluções entre os métodos numérico e analítico. Uma maneira de se mitigar tal diferença é através do ajuste do tamanho do passo, visto que, quanto menor o passo, menor será o erro cometido pela solução numérica.

10 Apêndice

A seguir, encontram-se alguns *links* para os *Notebooks Python* (.ipynb) utilizados para a confecção deste artigo.

- Exemplos iniciais de sistemas impulsivos: <https://colab.research.google.com/drive/15dVDbE09SzTWoQ1uII zgAqMt3IN?usp=sharing>
- Métodos de Runge-Kutta para a resolução de EDOs: <https://colab.research.google.com/drive/1YQ3UQDETlNpXME76OiCHA79FDtu9vJWC?usp=sharing>
- Método de Runge-Kutta Impulsivo para a resolução de EDIs: <https://colab.research.google.com/drive/1e-YuNvT6lViUP1kkvx4AJ5AK6ueDBk9J?usp=sharing>

Referências

STAMOVA, Ivanka; STAMOV, Gani. **Applied impulsive mathematical models**. Cham: Springer, 2016.

BONOTTO, Everaldo de Mello. **Sistemas semidinâmicos impulsivos**. 2005. 95 p. Dissertação (Mestrado em Matemática) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2005.

PATES, Richard. **The Lyapunov stability theorem**. [*S. l.: s. n.*], 2021. 1 vídeo (9 min). Disponível em: <https://www.youtube.com/watch?v=td-d4Yi-81c>. Acesso em: 2 jan. 2022.

PATES, Richard. **An example using the Lyapunov stability theorem**. [*S. l.: s. n.*], 2021. 1 vídeo (10 min). Disponível em: <https://www.youtube.com/watch?v=WNc7jWAKFTg>. Acesso em 2 jan. 2022.

RUGGIERO, Márcia A. Gomes; LOPES, Vera Lúcia da Rocha. **Cálculo numérico**: aspectos teóricos e computacionais. 2. ed. São Paulo: Pearson, 1996.

AZEVEDO, Anibal. **Projeto cálculo numérico para todos**. [*S. l.: s. n.*], 2021. playlist 142 vídeos. Disponível em: <https://www.youtube.com/playlist?list=PLH9knZH6lcgrCjPt7ouHphjuYvuzBfa3U>. Acesso em: 29 jul. 2022.