

**PESQUISA – MODELAGEM E SIMULAÇÃO CURSO DE
CIÊNCIA DA COMPUTAÇÃO UNIVERSIDADE
FRANCISCANA – UFN. 2025-02.**

PROFESSOR: André F. dos Santos.

Nome do aluno: José Otávio R. Baggio .

Data: 25 / 08 / 2025 .

Trabalho 01 – Pesquisa sobre Ferramentas de Modelagem e Simulação

Objetivo:

Investigar e analisar ferramentas computacionais atuais utilizadas em modelagem e simulação, identificando suas características principais, aplicações práticas e possíveis integrações com Inteligência Artificial.

Resumo:

Este trabalho tem como objetivo investigar ferramentas de modelagem e simulação utilizadas em diferentes contextos da Ciência da Computação e áreas correlatas. A atividade propõe a análise de suas características principais, classificações, formas de licenciamento, integração com técnicas de Inteligência Artificial e aplicações práticas no meio acadêmico e industrial.

Para atender aos requisitos da atividade, foram escolhidas três ferramentas distintas que representam diferentes paradigmas de simulação: **SymPy** e **Julia**.

I. SymPy:

a) Descrição Geral:

O SymPy é uma biblioteca escrita em Python para matemática simbólica. Seu principal objetivo é fornecer ferramentas para manipulação algébrica, resolução de equações, cálculo simbólico e análise matemática de forma simples e acessível. É muito utilizada em ensino, pesquisa e desenvolvimento de modelos matemáticos.

Entre suas funcionalidades estão: álgebra simbólica, cálculo diferencial e integral, resolução de sistemas de equações, séries, matrizes, polinômios, estatística, física e até geração de código em outras linguagens.



Figura 1: Imagem da Logotipo da biblioteca SymPy.

b) Classificação:

O SymPy é voltado para modelagem matemática contínua, já que lida principalmente com funções, equações e sistemas contínuos. No entanto, ele também pode ser aplicado em alguns problemas discretos, como teoria dos números e combinatória.

c) Licenciamento:

É um software open-source (licença BSD), totalmente gratuito, podendo ser utilizado tanto em ambiente acadêmico quanto em aplicações comerciais.

d) Recursos de Inteligência Artificial:

O SymPy não possui integração nativa com IA, mas pode ser facilmente combinado com bibliotecas de Machine Learning em Python, como TensorFlow, PyTorch e Scikit-learn, para auxiliar na formulação e resolução de problemas matemáticos que surgem em contextos de aprendizado de máquina.

e) Aplicações Práticas:

O SymPy é bastante usado em ensino de cálculo e álgebra, além de aplicações em ciência de dados, física computacional e engenharia. Um exemplo prático é a resolução de equações diferenciais que descrevem o comportamento de sistemas físicos, como crescimento populacional ou decaimento radioativo.

A seguir vemos o SymPy sendo utilizado na implementação de um cálculo algébrico, sendo esse o problema: Crescimento populacional exponencial

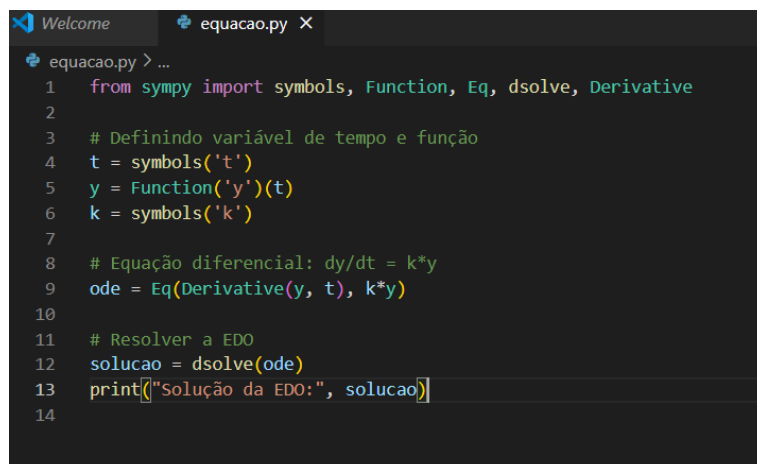
A equação diferencial que descreve o crescimento populacional sem limitações é:

$$\frac{dy}{dt} = k \cdot y$$

Aonde:

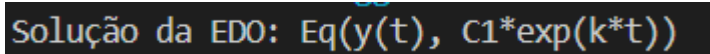
y = população em um tempo t,

k = taxa de crescimento.



```
Welcome equacao.py x
equacao.py > ...
1 from sympy import symbols, Function, Eq, dsolve, Derivative
2
3 # Definindo variável de tempo e função
4 t = symbols('t')
5 y = Function('y')(t)
6 k = symbols('k')
7
8 # Equação diferencial: dy/dt = k*y
9 ode = Eq(Derivative(y, t), k*y)
10
11 # Resolver a EDO
12 solucao = dsolve(ode)
13 print("Solução da EDO:", solucao)
14
```

Saída Esperada:



```
Solução da EDO: Eq(y(t), C1*exp(k*t))
```

II. Julia:



Figura 2: Logotipo da linguagem de Programação Julia.

a) Descrição Geral:

O Julia é uma linguagem de programação de alto desempenho desenvolvida para computação científica, simulação numérica e análise de dados. Seu grande diferencial é combinar a velocidade de linguagens compiladas (como C e Fortran) com a simplicidade de linguagens dinâmicas (como Python e MATLAB).

É muito usada em áreas como:

- Engenharia e simulação de sistemas físicos.
- Modelagem matemática e estatística.
- Inteligência Artificial e aprendizado de máquina.
- Big Data e computação de alto desempenho (HPC).

b) Classificação:

A Julia não é uma ferramenta de simulação pronta, mas sim uma linguagem de programação multimétodo, com pacotes que permitem modelagem contínua, discreta e estocástica.

Um dos pacotes mais famosos é o DifferentialEquations.jl, usado para resolver equações diferenciais, simulações de dinâmica de sistemas e modelos populacionais.

c) Licenciamento:

A Julia é open-source (licença MIT), gratuita para uso acadêmico, pesquisa e aplicações comerciais.

d) Recursos de Inteligência Artificial:

A Julia possui forte integração com IA e Machine Learning. Pacotes como Flux.jl e MLJ.jl permitem desenvolver redes neurais, algoritmos de aprendizado supervisionado e não supervisionado. Além disso, ela pode se integrar facilmente com Python e TensorFlow.

e) Aplicação Prática:

O Julia é bastante utilizado em pesquisa científica, modelagem de epidemias, análise financeira, física computacional e otimização matemática.

Exemplo simples com DifferentialEquations.jl

Problema: Resolver numericamente a equação diferencial:

$$\frac{dy}{dt} = y, y(0) = 1$$

```
using DifferentialEquations

# Definindo o problema diferencial
function f(du, u, p, t)
    du[1] = u[1]
end

u0 = [1.0]          # Condição inicial y(0) = 1
tspan = (0.0, 5.0) # Intervalo de tempo
prob = ODEProblem(f, u0, tspan)

# Resolver o problema
sol = solve(prob)

# Mostrar resultados
println(sol)
```

Isso retorna a solução numérica para a função $y(t) = e^t$, mostrando valores calculados entre 0 e 5.