Uso de software livre para modelagem matemática e simulação

Rafael de Oliveira Silva[1]
Sônia Ternes[2]

Um modelo matemático pode ser definido como um conjunto de símbolos e relações matemáticas que traduzem, de alguma forma, um fenômeno em questão (BASSANEZI, 1997). Uma vez representado o fenômeno estudado em linguagem matemática ou, mais especificamente, por meio de um sistema de equações diferenciais, torna-se possível prever o comportamento desse sistema por meio de simulações. Nesse contexto, surge a necessidade do uso de robustos softwares, capazes de computar, construir gráficos, resolver sistemas e equações, assim como gerar aproximações numéricas.

Existem no mercado uma infinidade de softwares matemáticos, sistemas computacionais numéricos e algébricos e ferramentas de simulação. Há produtos comerciais muito utilizados pela comunidade científica, como Mathematica e Matlab, e softwares não comerciais que podem ser utilizados como alternativa às ferramentas proprietárias. Além disso, as ferramentas não comerciais estão disponíveis sob uma licença de software livre (WEBER, 2004), possibilitando que qualquer usuário possa copiar, alterar e distribuir sem restrições, o que contribui para complementação e consequente melhoria da ferramenta, além de permitir a autonomia tecnológica.

[1] *Universidade Estadual de Campinas; rafaelos@cnptia.embrapa.br*

[2] *Embrapa Informática Agropecuária; sonia@cnptia.embrapa.br*

Este trabalho tem como objetivo avaliar o potencial de ferramentas de software livre para uso em modelagem matemática e simulação. Primeiramente foram pesquisadas e selecionadas as principais plataformas de softwares comerciais e não comerciais disponíveis para a comunidade científica. Os requisitos observados foram organizados numa tabela contendo, para cada software, nome, software livre equivalente/software pago equivalente, sistema operacional (Windows, Linux, MacOS), tipo de licença, objetivo principal, vantagem, desvantagem, forma de aquisição, observações gerais. Após ampla pesquisa, foram selecionados 46 softwares, sendo 20 disponíveis sob licença proprietária e 26 sob licença de software livre. Dentre esses foram escolhidas para avaliação as plataformas Maxima, Octave, Scilab e Sage. O modelo clássico Presa-Predador, proposto em 1926 pelo matemático Vito Volterra (MURRAY, 1989), que descreve a dinâmica da predação de uma espécie sobre outra através de um sistema de equações diferenciais, foi implementado em cada umas das ferramentas, e os resultados obtidos foram comparados. O modelo de Volterra é descrito pelo seguinte sistema de equações:

$$\frac{dx}{dt} = - 0.5x + 0.01xy, (predador)$$

$$\frac{dx}{dt} = 0.5y - 0.01xy, (presa)$$

Onde: x(t) é a população de predadores e y(t) a população de presas no instante t.

Condições iniciais: $x_0 = 80$; $y_0 = 100$;

Em todas as ferramentas avaliadas o procedimento se repetiu: 1) encontrar uma implementação do método de Runge-Kutta (desenvolvido por C. Runge e M.W. Kutta em 1900) para solução do sistema de equações; 2) plotar os dados, x(t) e y(t) em função do tempo, e 3) plotar o plano de fase x(t) versus y(t), a fim de observar a dinâmica presa-predador.

As Figuras de 1 a 4 mostram os resultados obtidos em cada plataforma. Os gráficos gerados foram exatamente os mesmos para o mesmo método numérico. Como o parâmetro “passo”, que controla a precisão do método de Runge-Kutta e a precisão numérica de ponto flutuante não foram controlados em nenhum dos softwares, a obtenção de gráficos idênticos sugere que os valores padrões para tais parâmetros são os mesmos em todas as plataformas. No que se refere à interface gráfica, a ferramenta mais “amigável” é o Octave (interface Qtoctave). Nele, as rotinas foram escritas utilizando-se a

Maxima

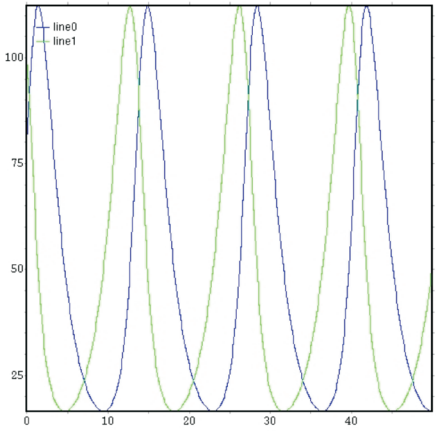

Figura 1a. Série temporal do sistema de Volterra.

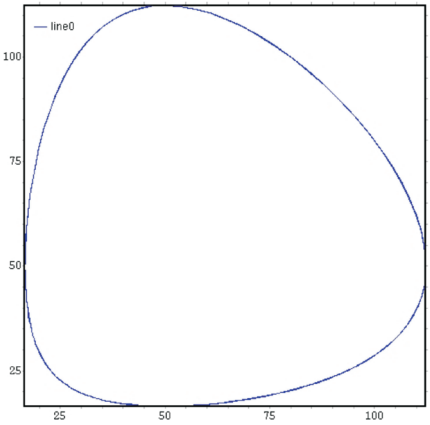

Figura 1b. Plano de fase do sistema de Volterra.

Octave

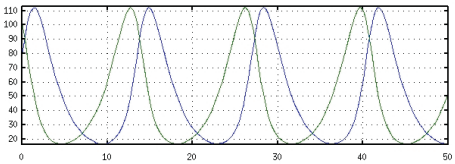

Figura 2a. Série temporal do sistema de Volterra.

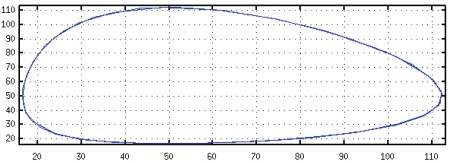

Figura 2b. Plano de fase do sistema de Volterra.

Scilab

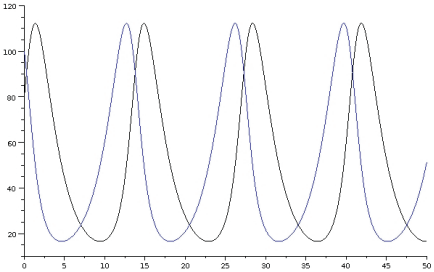

Figura 3a. Série temporal do sistema de Volterra.

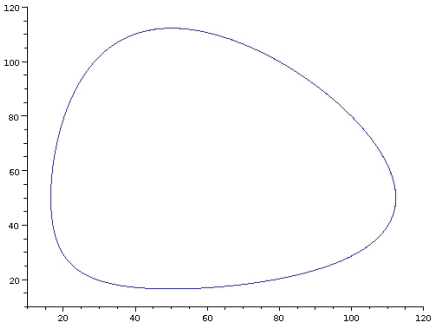

Figura 3b. Plano de fase do sistema de Volterra.

Sage

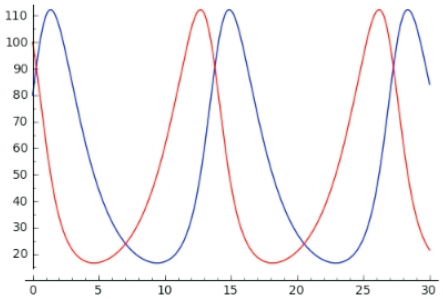

Figura 4a. Série temporal do sistema de Volterra.

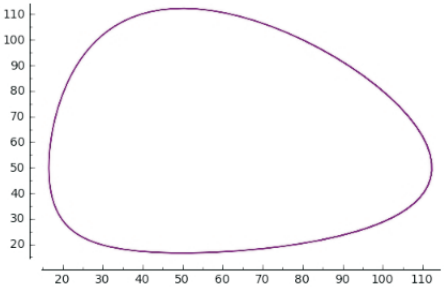

Figura 4b. Plano de fase do sistema de Volterra.

linguagem Matlab, justamente para testar a compatibilidade entre tais ferramentas, o que gerou uma única incompatibilidade: no Matlab a função que aplica o algorítimo de Runge-Kutta chama-se “ode45” e no Octave, “rk4”. A ferramenta que gerou mais dificuldades (“bugs”) de programação foi o Maxima. O Scilab, embora sua linguagem seja parecida com a do Matlab, tem seu desenvolvimento independente e não propõe ser um “clone” como o Octave. O Sage é a ferramenta mais robusta, por se tratar de uma integração de várias bibliotecas e pacotes de código aberto, porém, com a desvantagem de exigir pro-

ficiência em linguagem Python. As quatro ferramentas avaliadas são bastante robustas e possuem várias bibliotecas com aplicações em biomatemática.

Referências

BASSANEZI, R. **Modelagem Matemática**. Blumenau: Dynamis, 1997. 389 p.

MURRAY, J. D. **Mathematcal Biology**. Seattle: Springer, 1989. 767 p.

WEBER, S. **The sucess of Open Source**. Cambridge: Harvard University, 2004. 320 p.