**Proceeding Series of the Brazilian Society of Computational and Applied Mathematics**

# Uso da Computação Por Intervalos para Cálculo de Ponto Fixo de um Mapa Discreto

Heitor Magno Rodrigues Junior[1]

Programa de Pós-Graduação em Engenharia Elétrica, Modelagem e Controle de Sistemas, UFSJ, São João del-Rei, MG

Erivelton Geraldo Nepomuceno[2]

Departamento de Engenharia Elétrica, Modelagem e Controle de Sistemas, UFSJ, São João del-Rei, MG

**Resumo**. Neste trabalho, a aritmética intervalar é aplicada para o cálculo de ponto fixo de funções recursivas. Nesta abordagem, números são tratados como intervalos. Foi possível identificar a convergência correta para ponto fixo de período 1, tendo como estudo de caso o mapa logístico.

**Palavras-chave**. Computação por Intervalos, Mapa Logístico, Ponto Flutuante, Funções Recursivas

## 1 Introdução

A computação numérica é parte vital da infraestrutura científica moderna e é essencial em disciplinas científicas [4, 6]. Os computadores são usados para resolver equações que modelam os mais diferentes sistemas: desde a expansão do universo até a microestrutura de um átomo, processamento de imagens e análise de estatística de dados médicos, simulação de circuitos para projetos de computadores cada vez mais rápidos, menores e mais confiáveis, modelagem de aeronaves para testes e treinamento de pilotos e confiabilidade de sistemas elétricos, para citar alguns exemplos [10].

Praticamente toda a computação numérica utiliza a aritmética de ponto flutuante com grande parte dos computadores atuais fazendo uso da norma IEEE 754 [3, 5] para aritmética de ponto flutuante [10]. Porém, apesar da sua importância para a infraestrutura científica moderna e de apresentar resultados satisfatórios e muito próximos com os esperados, a computação numérica ainda está longe de ser uma ferramenta que disponibiliza resultados totalmente de acordo com a realidade. Isso acontece pois a aritmética de ponto flutuante é uma aproximação sistemática da aritmética dos números reais e é

[1]heitormrjunior@hotmail.com
[2]nepomuceno@ufsj.edu.br

representada por um subconjunto finito dos números reais. Consequentemente, algumas propriedades da aritmética dos números reais não são garantidas para a aritmética de ponto flutuante [5].

Devido a este fato, é importante chamar a atenção para as limitações dos computadores com respeito à computação científica, uma vez que a realização de experimentos no mundo físico ou cálculos advindos de alguma regra de contagem podem originar medições ou resultados parciais que precisem ser representados e analisados em computadores [9].

Como o computador apresenta limitações de armazenamento que fazem com que o conjunto de números que ele representa seja finito, cada número armazenado pelo computador representa um intervalo do conjunto dos números reais. Sendo assim, é importante levar em conta esses intervalos nas operações realizadas pela máquina para tratar o problema do erro nos computadores.

Problemas descritos por funções recursivas requerem o uso da computação numérica por se tratarem de problemas complexos em que o resultado da iteração atual sempre depende do anterior. Existe grande interesse no desenvolvimento de circuitos eletrônicos que representem o comportamento caótico dessas funções em diversas aplicações, como na geração de números aleatórios, comunicação em espectro de difusão em frequência variável [12] e geração de mapas caóticos, como o mapa logístico [7].

## 2 Propósito

O principal objetivo do artigo é apresentar uma solução para o erro computacional quando se trabalha com funções recursivas. Em [9], estabeleceram-se os princípios norteadores para o estudo da convergência de funções recursivas em computadores. Naquele trabalho, o autor calculou o erro por meio da análise da propagação do erro em cada operação aritmética, bem como, no cálculo impreciso da função recursiva. Neste trabalho, pretende-se obter o mesmo resultado por meio da aritmética intervalar [8].

De acordo com [9], as funções recursivas podem ser definidas da seguinte forma: seja $\mathbb{I} \subseteq \mathbb{R}$ um espaço métrico com $f : \mathbb{I} \longrightarrow \mathbb{R}$, tem-se que:

$$x_n = f(x_{n-1}). \tag{1}$$

Séries de tempo discreto podem ser geradas por um procedimento interativo simples de (1). Dependendo da escolha de $f$, a série pode apresentar tanto ponto fixo de período 1 ou maior, quanto comportamento caótico.

Segundo [1], se $f(x^*) = x^*$, então diz que $x^*$ é um ponto fixo de $f(x)$. O princípio do mapeamento da contração é um meio simples para encontrar o ponto fixo a partir de uma condição inicial com um número $x_0$ arbitrário e da definição da sequência $\{x_n\}$ por $x_n = f(x_{n-1})$ [2, 11]. Se essa sequência for convergente, então $x_n \to x^*$ à medida em que $n \to \infty$. Porém, em muitos casos, os cálculos são feitos por um computador e podem apresentar erros.

Um número real é armazenado no computador como ponto flutuante, que é um formato de representação digital de números reais usado nos computadores. O ponto flutuante é usado em computadores desde meados da década de 1950, onde durante as duas décadas

seguintes cada empresa adotava um padrão diferente. Isso acarretava em incompatibilidade na forma como um programa se comportava em máquinas diferentes. Buscando uma solução ao problema, nas décadas de 1970 e 1980, um grupo de cientistas e engenheiros, liderados pelo *Institute for Electrical Engineering and Electronics Engineers – IEEE*, desenvolveu um padrão para a representação e aritmética de ponto flutuante [10].

O computador apresenta um limite de armazenamento de bits, que são usados para representar os dados de cada componente das operações computacionais, restringindo a quantidade de números reais que podem ser armazenados, dependendo do tipo do formato adotado. Isso faz com que a máquina trate números como intervalos. A distância entre um número $x$ armazenado no computador em relação a seu antecessor e sucessor que também podem ser armazenados é chamada de *ulp*. Nenhum dos números existentes entre os números $x$ e $y = x + \frac{1}{2}ulp(x)$ podem ser armazenados pela máquina, então seus valores convergem para um desses dois números. Quanto maior o valor de $x$, maior também é sua *ulp*, ou seja, maior é o intervalo de valores que não são representados pelo computador. Isso faz com que a representação dos números em ponto flutuante produza erros de arredondamento que se propagam ao longo das operações que executam um algoritmo [10].

A técnica de computação por intervalos foi proposta em sua forma moderna por [8] como uma ferramenta para limitar os erros de arredondamentos na computação numérica e é um campo amplo que associa a matemática rigorosa com a computação científica. É uma aritmética que define conjuntos de intervalos em vez de números reais. A ligação entre a computação e a matemática intervalar torna possível resolver problemas que não podem ser resolvidos de forma eficiente usando a aritmética tradicional de ponto flutuante [8].

Por um intervalo, entende-se um conjunto fechado e limitado de números reais. Pode-se considerar também o intervalo como um par ordenado de pontos limites, denotado por uma letra maiúscula [8]. Além disso, dados dois intervalos denotados por $X$ e $Y$, se $X \cap Y \neq \emptyset$, não se pode afirmar que $X$ é diferente de $Y$ [8].

A função recursiva usada para ilustrar a técnica de computação por intervalos na eliminação do erro computacional é a equação do mapa logístico, dada por:

$$x_{n+1} = rx_n(1 - x_n). \tag{2}$$

Essa equação foi desenvolvida por [7] como um modelo populacional, com $x_n$ sendo um número entre zero e um que representa a razão entre a população existente na n-ésima geração e o maior número possível de indivíduos e $r$ como sendo uma taxa de crescimento da população. Escolhendo um valor para o parâmetro $r$ e iterando recursivamente o mapa a partir de uma condição inicial $x_0$, obtém-se uma série temporal da equação do mapa logístico.

## 3 Métodos

O algoritmo da Figura (1) pode ser desenvolvido para a representação da equação (2) com o parâmetro $r = 327/100$ e condição inicial $x_0 = 100/327$. O resultado é mostrado na Figura (2), onde está claro que a simulação converge para pontos fixos de período 2.

**Algorithm 1**

```
r ← 327/100;
x_0 ← 100/327;
N ← 200;
for n ← 0 to N do
    x_{n+1} ← r x_n (1 − x_n);
end
```

Figura 1: Algoritmo Básico da Equação Logística. Fonte: [9].

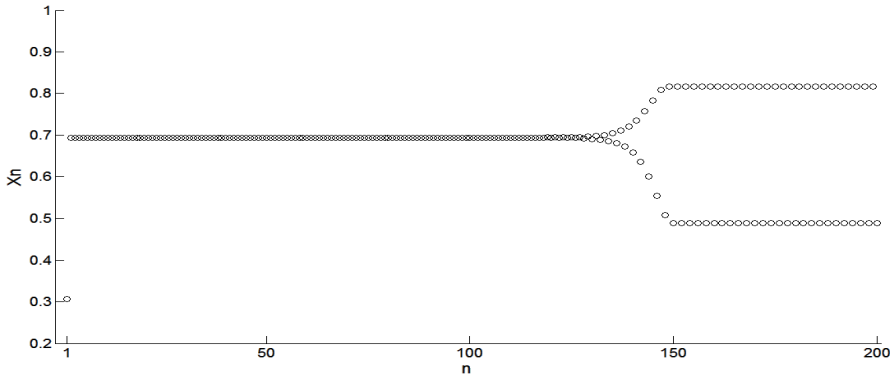


Figura 2: Simulação de (2) com $r = 327/100$ e $x_0 = 100/327$.

Entretanto, este não é o resultado correto. Através de uma análise simples, pode-se chegar à conclusão de que a equação do mapa logístico para os parâmetros especificados só poderia convergir para um ponto fixo. O resultado pode ser facilmente demonstrado por:

$$x_1 = \frac{\cancel{327}}{\cancel{100}}\frac{\cancel{100}}{\cancel{327}}\left(1 - \frac{100}{327}\right) = \frac{327 - 100}{327} = \frac{227}{327}.$$

$$x_2 = \frac{\cancel{327}}{100}\frac{227}{\cancel{327}}\left(1 - \frac{227}{327}\right) = \frac{227}{\cancel{100}}.\frac{\cancel{100}}{327} = \frac{227}{327}.$$

Portanto, tem-se que $x_1 = x_2 = ... = x_n = 227/327$, o que significa que a equação (2) para os parâmetros especificados converge para o ponto fixo $x = 227/327$, invalidando o resultado obtido pelo computador, que não pode representar precisamente os parâmetros e reproduzir fielmente os resultados das operações dos números reais. Isso faz com que erros sejam acumulados a cada iteração que passa, levando a um resultado totalmente divergente da realidade. A solução encontrada para eliminar o erro computacional foi trabalhar com intervalos ao invés de pontos, tornando as operações mais maleáveis e levando em consideração os erros da máquina.

De acordo com [9], pode-se escrever uma versão adaptada a aritmética intervalar para a definição de ponto fixo.

**Definição 1.** *Sejam* $f(x^*) = [\underline{f}(x^*),\overline{f}(x^*)]$ *e* $x^* = [\underline{x}^*,\overline{x}^*]$. *Se* $x^* \cap f(x^*) \neq \emptyset$, *então* $x^*$ *é ponto fixo.*

Na intenção de solucionar o problema de falta de precisão de dados da máquina, foi desenvolvido um algoritmo que leva em conta os intervalos de representação dos números no computador e tem como objetivo descobrir se uma condição inicial $x_0$ direciona a solução para um ponto fixo $x^*$ em alguma iteração $n$.

Para representar a largura de um intervalo, utilizou-se a *ulp* do número. Assim, para uma condição inicial $x_0$, os limites do intervalo foram dados arredondando $x_0$ para baixo e para cima, de modo que o computador não conseguisse reproduzir nenhum número que estivesse dentro desse intervalo. A partir disso, as operações são feitas considerando esses limites e o resultado final é dado pelo ponto médio dos intervalos resultantes. Foi usado o algoritmo base representado na Figura (3) para calcular a resposta da equação (2) para $x_0 = 100/327$ com o parâmetro $r = 327/100$.

**Algorithm 2**

$r \leftarrow 327/100$;
$x_0 \leftarrow 100/327$;
$N \leftarrow 200$;
$x_0^- \leftarrow round^-(x_0)$;
$x_0^+ \leftarrow round^+(x_0)$;
**for** $n \leftarrow 0$ **to** $N$ **do**
  $x_{n+1}^- \leftarrow r\, x_n^-(1 - x_n^-)$;
  $x_{n+1}^+ \leftarrow r\, x_n^+(1 - x_n^+)$;
  $X_n = [x_n^-, x_n^+]$;
  $X_{n+1} = [x_{n+1}^-, x_{n+1}^+]$;
  **if** $X_n \cap X_{n+1} \neq \emptyset$
    $x_n^* = \dfrac{x_n^- + x_n^+}{2}$ é ponto fixo;
    **break;**
  **end**
**end**

Figura 3: Algoritmo base para cálculo de ponto fixo.

## 4 Resultados

Os resultados obtidos como solução do algoritmo da Figura (3) para o parâmetro $r = 327/100$ e a condição inicial $x_0 = 100/327$ são apresentados na tabela (1). A tabela conta com os valores, a cada iteração $n$, do ponto médio do intervalo $x_n$, do tamanho do intervalo definido por $[x_n^-, x_n^+]$ e do tamanho da interseção entre o intervalo atual e o intervalo da iteração anterior. De acordo com o algoritmo da Figura (3), quando os intervalos que representam duas iterações consecutivas apresentarem interseção diferente de conjunto vazio, então o algoritmo é parado e o valor do ponto fixo $x^*$ é dado por $x_n$.

Tabela 1: Resultados obtidos como solução do algoritmo da Figura (3).

| $n$ | $x_n$ | $\lvert x_n^+ - x_n^- \rvert$ | $\lvert X_n \cap X_{n-1} \rvert$ |
|---|---|---|---|
| 0 | 0,305810397553517 | $5{,}551115123125783 \times 10^{-17}$ | $= \emptyset$ |
| 1 | 0,694189602446483 | $3{,}330669073875470 \times 10^{-16}$ | $= \emptyset$ |
| 2 | 0,694189602446483 | $2{,}220446049250313 \times 10^{-16}$ | $\neq \emptyset$ |

Pela análise da tabela, pode-se observar que o ponto fixo é atingido na segunda iteração, uma vez que $|X_n \cap X_{n-1}| \neq \emptyset$. Assim, adota-se o valor para o ponto fixo como $x^* = 0{,}694189602446483$. É importante destacar que o valor de $X_n \neq X_{n-1}$ e que se houver continuidade da simulação computacional, pode-se atingir uma região de estabilidade equivocada, conforme já foi apontado em [9].

## 5 Conclusão

Pode-se observar que, mesmo com intervalos de tamanhos muito pequenos em relação a seus pontos médios, já na segunda iteração existe interseção entre os intervalos atual e anterior. Além disso, o intervalo de interseção tem o mesmo tamanho do intervalo dessa iteração. Isso mostra que o intervalo da iteração 2 é subconjunto do intervalo da iteração anterior.

A partir do momento em que a interseção entre os intervalos de iterações consecutivas é diferente de conjunto vazio, não se pode afirmar que são resultados diferentes. E com isso, prosseguir resolvendo o programa pelo computador pode resultar em inconsistência matemática na resposta, fazendo com que resultados sejam invalidados.

O caso específico apresentado pode ser estendido para todas as outras condições iniciais que respeitem as restrições do problema e para todos os valores possíveis do parâmetro da taxa de crescimento da população.

Apesar de sua importância e capacidade única em realizar cálculos e operações com a rapidez que nenhum ser humano possui, o computador não é uma máquina perfeita. Portanto, apresenta falhas em seu sistema que devem ser levadas em consideração por todos os usuários. Essas falhas são devidas às limitações físicas que levam a uma falta de precisão no armazenamento dos números. Essa falta de precisão leva o computador a práticas que podem interferir consideravelmente na resposta do sistema.

Finalmente, antes de tomar como verdade os resultados obtidos pelo computador, é importante saber como os números reais são tratados na máquina e deve-se levar em conta a forma como o computador lida com as operações matemáticas e arredondamentos. Os resultados aqui obtidos são coerentes com aqueles obtidos em [9], mas apresentam uma maior simplicidade matemática. Pretende-se em trabalhos futuros expandir o conceito apresentado ampliando esses resultados para outros mapeamentos e realizando a mesma análise para o caso de pontos fixos de períodos superiores.

## Agradecimentos

Agradecemos à CAPES, CNPq/INERGE, FAPEMIG e à Universidade Federal de São João del-Rei pelo apoio.

## Referências

[1] M. J. Feigenbaum. Quantitative Universality for a Class of Non-linear Transformations. *journal Of Statistical Physics*, 19(1):25–52, 1978.

[2] W. L Ferrar. *A Text-Book of Convergence*. Oxford : Clarendon Press, 1938.

[3] D. Goldberg. What every computer scientist should know about floating-point arithmetic. *Computing Surveys*, 23(1):5–48, March 1991.

[4] S.M. Hammel, J.A. Yorke, and C. Grebogi. Do numerical orbits of chaotic dynamical processes represent true orbits? *Journal of Complexity*, 3(2):136–145, 1987.

[5] Institute of Electrical and Electronics Engineers (IEEE). *754-2008 – IEEE standard for floating-point arithmetic*. IEEE, 2008. p. 1–58.

[6] René Lozi. Can we trust in numerical computations of chaotic solutions of dynamical systems? In *Topology and Dynamics of Chaos: In Celebration of Robert Gilmore's 70th Birthday. Edited by Letellier Christophe. Published by World Scientific Publishing Co. Pte. Ltd., 2013. ISBN# 9789814434867, pp. 63-98*, volume 1, pages 63–98, 2013.

[7] R. M. May. Simple mathematical models with very complicated dynamics. *Nature*, 261:459–467, 1976.

[8] Ramon E Moore and R. E Moore. *Methods and applications of interval analysis*, volume 2. SIAM, 1979.

[9] Erivelton Geraldo Nepomuceno. Convergence of recursive functions on computers. *The Journal of Engineering*, pages 1–3, October 2014.

[10] M. L. Overton. *Numerical Computing with IEEE floating point arithmetic*. SIAM, 2001.

[11] Walter Rudin. *Principles of mathematical analysis*. International Student Edition. McGraw-Hill New York, 3 edition, 1976.

[12] M. Suneel. Electronic circuit realization of the logistic map. *Sadhana-academy Proceedings In Engineering Sciences*, 31:69–78, February 2006.