UNIVERSIDADE FEDERAL DO PAMPA

Rafael Fogliato Alves

Definição da função densidade de probabilidade com distribuição Weibull intervalar e aplicação de diferentes métodos de integração

Alegrete
2017

Rafael Fogliato Alves

Definição da função densidade de probabilidade com distribuição Weibull intervalar e aplicação de diferentes métodos de integração

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Ma. Alice F. Finger

Alegrete
2017

Rafael Fogliato Alves

Definição da função densidade de probabilidade com distribuição Weibull intervalar e aplicação de diferentes métodos de integração

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Trabalho de Conclusão de Curso defendido e aprovado em 29 de junho de 2017
Banca examinadora:

Alice Fonseca Finger

Prof. Ma. Alice F. Finger
Orientadora
UNIPAMPA

Prof. Dr. Marcelo Resende Thielo
UNIPAMPA

Prof. Me. Celso Nobre da Fonseca
UNIPAMPA

Este trabalho é dedicado às crianças adultas que,
quando pequenas, sonharam em se tornar cientistas.

AGRADECIMENTOS

Agradeço primeiramente a minha família, principalmente à minha mãe e irmão, por todo o suporte, carinho e ajuda, sem os quais provavelmente nem estaria fazendo esse trabalho. Agradeço também aos Inocentes, cuja amizade e companheirismo, construídos durante toda nossa jornada juntos nessa etapa de nossas vidas foi de enorme importância pra mim, e sem os quais nem sei se não teria desistido na primeira dificuldade encontrada. Então muito obrigado, Amanda, Bolívar, Murilo, Luana, Paloma, Priscila e Romário. Por toda ajuda, de imensurável importância (e por toda a paciência), agradeço à Prof. Alice, esse trabalho só existe devido a sua ótima orientação Agradeço também a todos que ajudaram de qualquer forma nesse período de graduação, afinal, sozinho não teria conseguido fazer nada. Obrigado.

“Hmmm... o resultado não é tudo o que procuro.
Quando a única coisa que importa é o resultado,
você começa a tentar tomar atalhos...
e quando se toma atalhos demais,
você pode perder de vista o que realmente importa.
E, eventualmente, perderá a sua motivação também.
(Hirohiko Araki; JoJo’s Bizarre Adventures)

RESUMO

Quando são realizados cálculos em um sistema computacional, trabalha-se com números de ponto flutuante, o que faz com que não seja possível representar os valores corretamente. Devido a isso, o resultado dos cálculos é sujeito a erros de aproximação, como os erros de arredondamento e de truncamento. Uma das alternativas de contornar os erros causados pela máquina é a utilização da aritmética intervalar, a qual garante intervalo solução onde o valor real está contido. Para obter o valor numérico da função densidade de probabilidade da variável aleatória contínua com distribuição Weibull, é necessária a resolução de integrais, nas quais o resultado é obtido através de aproximação, o que torna-o sujeito a erros como os de arredondamento e truncamento. Nesse trabalho define-se a função densidade de probabilidade com distribuição Weibull de forma intervalar, usando o método da extensão intervalar. Após definida, é feita a implementação da função com distribuição Weibull utilizando diferentes métodos de resolução de integral intervalar, já existentes na literatura, a fim de analisar qual método retorna um intervalo solução de melhor qualidade. Para verificar os resultados obtidos, será feita a análise de erro através dos cálculos de erro absoluto e relativo. Além disso, também foi feita analisada a complexidade dos algoritmos implementados, a fim de certificar que o esforço computacional ao usar aritmética intervalar é semelhante ao esforço despendido com cálculos com entradas reais. Com os resultados obtidos pôde-se determinar que o método de resolução intervalar que obteve os melhores resultados foi a primitiva intervalar da funcão com distribuição Weibull, enquanto o método de integração intervalar que mostrou os melhores valores foi o método de Simpson intervalar.

Palavras-chave: Aritmética intervalar. Distribuições de probabilidade. Integrais intervalares. Análise de erros.

ABSTRACT

When calculations are performed in a computational system, floating-point numbers are used, that makes with it isn't possible to represent the values correctly. One of the alternatives to avoid errors caused by the machine is the use of interval arithmetic, which guarantees a interval solution where the real value is contained. To obtain the numerical value of probability density functions of continuous random variable with Weibull distribution, is required the solution of integrals, in which the solution is obtained through aproximation, which makes it subject to errors, such as rounding and truncation errors. This work intends to define the probability density funcion with Weibull distribution in interval form, using the interval extension method. After the definition is obtained, the function with Weibull distribution was implemented using different methods to solve interval integrals already existing in literature, in order to analise which method returns a better quality solution interval. To verify the obtained results, a error analysis was done through the absolute and relative error. Furthermore, the implemented algorithm complexity will also be analyzed, in order to verify the computational effort when using interval arithmetic is similar to the effort spent with the calculations using real entrances. With the results, it was possible to determinate that the method to solve interval integrals with the best results was the interval primitive of the function, while the interval integration method that showed better results was the Interval Simpson method.

Key-words: Interval arithmetic. Probability distributions. Interval integrals. Error analysis

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O conjunto de números representáveis em qualquer máquina é finito, assim as operações numéricas realizadas em sistemas computacionais podem gerar vários tipos de erros, como, por exemplo, erros de truncamento, de arredondamento, propagação dos erros de dados de entrada, entre outros. Esses erros ocorrem quando: um valor real de entrada é aproximado para um número real de máquina (erro de truncamento e/ou arredondamento); os erros nos resultados intermediários gerados na execução de cada operação vão se acumulando (LORETO, 2006).

A aritmética intervalar é uma teoria matemática desenvolvida principalmente por Moore (1976) que tem como foco a manipulação e operação de intervalos para controlar, rigorosa e automaticamente, os diversos tipos de erros encontrados nos resultados da computação numérica (MESQUITA, 2004). Na matemática intervalar são usados intervalos reais para representar valores infinitos, desconhecidos ou contínuos, ou seja, com os intervalos é possível representar valores inexatos, aproximações e erros de truncamento (LORETO, 2006).

Na aritmética intervalar, ao invés de aproximar um valor real x para um valor de máquina, ele é 'contido' em um intervalo fechado **x** que possui limites superior e inferior, ou seja, é criado um intervalo $\mathbf{x} = [\underline{x}, \bar{x}]$, onde está presente o valor real x. O tamanho do intervalo, que é a diferença entre os limites superior e inferior, é uma das maneiras de avaliar a qualidade do intervalo solução, ou seja, quanto menor o tamanho do intervalo melhor é sua qualidade (RATSCHEK; ROKNE, 1988).

Os intervalos podem ser aplicados nas mais diversas áreas, tais como: programação matemática, manipulação de equações, análise e projeto de circuitos elétricos, psicologia matemática, estatística, equações diferenciais, física, entre outras (KEARFOTT, 1996).

Quando o uso de variáveis aleatórias contínuas sobre valores reais $\mathbb{R}$ é abordado, um dos problemas enfrentados é o cálculo de probabilidades, já que é necessária a resolução de uma integral definida da função densidade de probabilidade, para a qual nem sempre é possível obter a primitiva ou as vezes é difícil de se obter. Como integrais de funções densidade de probabilidade são resolvidas analiticamente, seu resultado é encontrado através de aproximação, o que o torna sujeito a erros como os de arredondamento e de truncamento (FINGER, 2014).

Os problemas numéricos resultantes da computação de probabilidade originam-se da impossibilidade de operação com números reais de forma direta, já que é preciso representar a reta real (uma grandeza contínua) em palavras de máquina (a qual representa valores discretos) (CAMPOS, 1997).

Para implementar funções intervalares em sistemas computacionais deve-se usar linguagens de programação que tenham o tipo intervalo definido, permitindo assim a manipulação dos intervalos e operações sobre esse tipo. Exemplos de linguagens que permitem o uso de intervalos são: C-XSC, Fortran-XSC, IntLab e IntPy.

Dentre as funções densidade de probabilidade com distribuição existentes na literatura, para variáveis aleatórias contínuas, algumas já foram definidas para entradas intervalares, como, por exemplo, funções densidade com distribuição Normal e Pareto.

A fim de complementar o que já tem na literatura, nesse trabalho será definida a função densidade de probabilidade com distribuição de Weibull de forma intervalar. Tal distribuição é frequentemente usada em questões referentes a confiabilidade, taxas de falhas, medidas de durabilidade e outras áreas devido a sua versatilidade e simplicidade(NAVIDI, 2016).

Também serão comparados os vários métodos para o cálculo de integrais intervalares, a fim de escolher qual tem melhor aplicação na forma intervalar da função densidade de probabilidade, retornando os melhores resultados e com melhor eficiência. Para implementação da função densidade de probabilidade será utilizado o pacote de extensão para programação intervalar da linguagem de programação Python, o IntPy (BARRETO, 2016), o qual permite a manipulação e operação do tipo intervalo.

1.1 Objetivos

O principal objetivo desse trabalho é definir com entradas intervalares a função densidade de probabilidade com distribuição Weibull. Essa definição será obtida através do uso das propriedades da matemática intervalar. A fim de atingir o objetivo principal, os objetivos específicos a serem executados são listados e detalhados logo abaixo:

- Definir a função densidade de probabilidade da distribuição Weibull com entradas intervalares: Na literatura encontram-se as definições das funções densidade de probabilidade das distribuições Uniforme, Exponencial, Normal, Beta, Gama e Pareto, assim, para completar as definições intervalares para as funções densidade de probabilidade das variáveis aleatórias será definida a forma intervalar da distribuição Weibull, através do método de extensão intervalar.

- Escolha de melhor método na resolução de integrais intervalares: No cálculo da probabilidade são resolvidas integrais definidas da função densidade de probabilidade. Como no presente trabalho será usada a aritmética intervalar, será necessário aplicar um método adequado a resolução de integrais com intervalos. Serão analisados os diversos métodos de integração intervalar disponíveis na literatura e, comparando os resultados retornados e questões referentes a complexidade computacional, será escolhido o método que melhor atender aos objetivos do trabalho.

- Analisar a qualidade do intervalo encapsulador: O intervalo encontrado para a função densidade de probabilidade da distribuição Weibull será analisado usando as medidas de erro absoluto e relativo e o cálculo de diâmetro do intervalo, para assim mostrar a qualidade do intervalo solução.

- Analisar a complexidade computacional: Os algoritmos implementados no trabalho terão sua complexidade analisada, a fim de investigar o esforço computacional gasto no cálculo da função densidade de probabilidade com distribuição na forma intervalar. A complexidade dos algoritmos para a distribuição em forma real e em forma intervalar serão comparados para comprovar que o esforço computacional para realizar o cálculo utilizando aritmética intervalar é semelhante ao esforço despendido quando é utilizada a forma real da função.

1.2 Organização do trabalho

Esse trabalho é organizado da seguinte forma:

No capítulo 2 é vista a fundamentação teórica sobre matemática intervalar, apresentando seus principais conceitos, operações básicas na aritmética intervalar e o método de extensão intervalar. Também são apresentados vários métodos para calcular integrais intervalares, mostrando suas características e fórmulas. Além disso é apresentado o pacote IntPy (BARRETO, 2016), a teoria da probabilidade e as métricas da qualidade do intervalo.

No capítulo 3 são apresentados trabalhos, já desenvolvidos, na área da matemática intervalar e probabilidade, os quais são utilizados neste trabalho como fonte de conceitos importantes.

O capítulo 4 trata sobre o desenvolvimento deste trabalho, onde são mostrados conceitos teóricos, os quais foram de grande importância no desenvolvimento desta monografia. É feita a definição intervalar da funcão densidade de probabilidade com distribuição Weibull e também são mostrados os resultados obtidos na implementação da forma intervalar da função, os quais são analisados na seção 4.2. Também é feita a análise de complexidade dos métodos, a qual é vista na seção 4.3.

O capítulo 5 apresenta as conclusões obtidas nesse trabalho, onde é mostrado o método para cálculo da forma intervalar da função que obteve os melhores resultados de maneira mais eficiente.

Por fim são mostradas as referências bibliográficas presentes nesta monografia.

2 FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo serão apresentados os conceitos e definições da matemática intervalar importantes para o entendimento e desenvolvimento do trabalho, bem como as operações aritméticas básicas (soma, subtração, multiplicação e divisão). Também será explicado o método da extensão intervalar, o qual é utilizado na definição da forma intervalar da função densidade de probabilidade das variáveis aleatórias contínuas com a distribuição Weibull. Além disso, será abordada a teoria da probabilidade com o objetivo de apresentar tanto os conceitos básicos das distribuições de probabilidade para variáveis aleatórias contínuas como a teoria da probabilidade intervalar definida por Campos (1997).

Por fim, será apresentado o ambiente computacional utilizado no trabalho, o pacote IntPy, o qual é adicionado à linguagem de programação Python para implementar a função densidade de probabilidade das variáveis aleatórias contínuas com distribuição Weibull na sua forma intervalar.

2.1 Matemática Intervalar

A aritmética intervalar (MOORE, 1966) (MOORE, 1979) (MOORE; KEARFOTT; CLOUD, 2009) tem como base o uso de intervalos fechados $[\underline{x}, \bar{x}]$ de valores reais como seus elementos básicos. No ponto de vista computacional sua definição se dá na seguinte maneira: dada uma função f(x) de variável real x pertencente a um intervalo $\mathbf{x} = [\underline{x}, \bar{x}]$, onde $\underline{x}, \bar{x} \in \mathbb{R}$, a imagem da função é dada por:

$$f(x) = \{y | y = f(x), \underline{x} \leq \mathbf{x} \leq \bar{x}\},$$

onde é possível determinar um intervalo $\mathbf{y}$, no qual $\mathbf{y} = [\underline{y}, \bar{y}]$, tal que $f(x) \subseteq \mathbf{y}$, ou seja, a função $f(x)$ está encapsulada no intervalo $\mathbf{y}$. Partindo disso é possível definir uma função F associada a f através da transformação do intervalo $\mathbf{x}$ no intervalo $\mathbf{y}$, ficando assim:

$$f(x) \subseteq F(x) = \mathbf{y}.$$

A função F, chamada de extensão intervalar de f, deve possuir a menor diferença possível da imagem $f(x)$. A matemática intervalar usa um tipo especial de arredondamento, o arredondamento direcionado, onde os resultados são arredondados para o menor e maior número de máquina que contenham o resultado das operações, o que resulta em um intervalo de máquina que contém a solução do cálculo.

Na extensão intervalar, o programa f é implementado através da substituição de cada operação elementar de números reais por operações correspondentes usando aritmética intervalar. A extensão intervalar é definida da seguinte maneira:

A função $F : \mathbb{I}(\mathbb{R}) \to \mathbb{I}(\mathbb{R})$ é uma extensão intervalar de uma função $f : \mathbb{R} \to \mathbb{R}$, se, para todo $x \in \mathbb{R}$, $f([x, x]) = [f(x), f(x)]$ (SANTIAGO; BEDREGAL; ACIÓLY, 2006).

Quando se trata de aritmética intervalar é importante diferenciar os conceitos de imagem intervalar de uma função e de avaliação intervalar. A imagem intervalar de uma função f, contínua no intervalo $\mathbf{x}$, é definida como sendo o intervalo limitado pelo mínimo e pelo máximo da imagem $f(x)$, sendo x um elemento encapsulado pelo intervalo $\mathbf{x}$, o qual possui a seguinte notação:

$$Im(f(x)) = \{min[f(x)], max[f(x)] | x \in \mathbf{x}\}$$

Na computação intervalar o intervalo solução $\mathbf{y} = [\underline{y}, \bar{y}]$ pode ser obtido através do uso de métodos de aproximação, técnicas de otimização, extensão intervalar, entre outros (KREINOVICH, 2003). No princípio da aritmética intervalar problemas como a não-portabilidade de códigos que implementassem essa aritmética e o esforço extra imposto pela computação intervalar ao processador e a memória faziam com que o uso da matemática intervalar não fosse muito atrativo (HICKEY; ENDEN, 1999).

A extensão intervalar (MOORE, 1976), também chamada de avaliação intervalar (OLIVEIRA; DIVERIO; CLAUDIO, 1997), é definida como o método primário na computação do intervalo solução (FERSON et al., 2004). Esse método se baseia no fato de que, em um computador, todo algoritmo consiste de operações elementares (aritméticas e lógicas). Para todas as operações elementares $f(a, b)$, caso sejam conhecidos os intervalos de a e b (chamados de $\mathbf{A}$ e $\mathbf{B}$, respectivamente), é possível computar a imagem de $f(\mathbf{A}, \mathbf{B})$ usando a aritmética intervalar definida por Moore (1966).

2.1.1 Operações aritméticas básicas

Na matemática intervalar é possível realizar o cálculo de valores através de operações elementares da matemática básica como adição, subtração, multiplicação e divisão. Mas, ao invés de operar sobre valores reais, os cálculos são feitos com intervalos.

Todas as operações aritméticas básicas que serão explicadas utilizarão como elementos os intervalos $\mathbf{x} = [\underline{x}, \bar{x}]$ e $\mathbf{y} = [\underline{y}, \bar{y}]$.

- Adição: A soma entre dois intervalos é apresentada como:

  $$\mathbf{x} + \mathbf{y} = [\underline{x} + \underline{y}, \bar{x} + \bar{y}].$$

  Ex.: Dados os intervalos $\mathbf{x} = [2, 4]$ e $\mathbf{y} = [1, 3]$, o resultado da soma entre eles resultará no intervalo $\mathbf{x} + \mathbf{y} = [3, 7]$.

- Subtração: A diferença entre dois intervalos é calculada:

  $$\mathbf{x} - \mathbf{y} = [\underline{x} - \underline{y}, \bar{x} - \bar{y}].$$

  Ex.: Dados os intervalos $\mathbf{x} = [2, 4]$ e $\mathbf{y} = [1, 3]$, o resultado da subtração entre eles resultará no intervalo $\mathbf{x} - \mathbf{y} = [1, 1]$.

- Multiplicação: O produto entre dois intervalos é apresentado como:

$$\mathbf{x} \cdot \mathbf{y} = [\underline{x} \cdot \underline{y}, \bar{x} \cdot \bar{y}].$$

Ex.: Dados os intervalos $\mathbf{x} = [2, 4]$ e $\mathbf{y} = [1, 3]$, o resultado da multiplicação entre eles resultará no intervalo $\mathbf{x} \cdot \mathbf{y} = [2, 12]$.

- Divisão: O quociente entre dois intervalos é apresentado como:

$$\mathbf{x} \div \mathbf{y} = [\underline{x} \div \underline{y}, \bar{x} \div \bar{y}].$$

Ex.: Dados os intervalos $\mathbf{x} = [2, 4]$ e $\mathbf{y} = [1, 2]$, o resultado da divisão entre eles resultará no intervalo $\mathbf{x} \div \mathbf{y} = [2, 2]$. Assume-se que o intervalo $\mathbf{y}$ não possui elementos nulos, ou seja $\underline{y} \neq 0$ e $\bar{y} \neq 0$.

2.2 Integrais intervalares

Na literatura existem diferentes maneiras de resolução de integral utilizando intervalos. Nesta subseção serão apresentados os mais utilizados e que serão adotados no desenvolvimento do trabalho.

Dada uma função $f : \mathbb{R} \to \mathbb{R}$ no intervalo $A = [a, b]$ deseja-se computar a seguinte integral:

$$T = \int_a^b f(x)\, dx$$

Nesse problema, caso seja encontrada uma função F tal que $F(x) = f(x)$, então T é simplesmente calculado como $T = f(b) - f(a)$. Porém, caso não seja encontrada essa função, ou caso sua implementação seja indisponível ou muito complexa, o resultado é obtido através de aproximações, o que deixa o cálculo da integral sujeito a erros, como os de arredondamento e truncamento (NOBREGA, 2010).

Em vista disso foram propostos vários métodos para calcular integrais utilizando aritmética intervalar, os quais serão vistos abaixo.

- **Integral de Moore**: No estudo da aritmética intervalar feito por Moore., Yang e Strother (1960), a integral $\int_A F(\mathbf{x})dx$ era definida por funções F, onde o domínio e alcance são contidos em intervalos fechados e delimitados de números reais. Nesse método o intervalo é dividido em n subconjuntos igualmente espaçados, onde tem-se:

$$S = \{a = x_0, x_1, x_2, ..., x_{n-1}, b = x_n\},$$

para $[a, b]r$, onde r é um número real, ou seja, o conjunto de todos os valores xr de maneira com que $x \in [a, b]$. Isso é visto no somatório:

$$\sum(F,S) = F[x_0,x_1](x_1 - x_0) + F[x_1,x_2](x_2 - x_1) + ... + F[x_{n-1},x_n](x_n - x_{n-1})$$

Assim, a integral de Moore é definida por cada função contínua de todos os intervalos [**A**,**B**] para cada $F(\mathbf{A}) \in F(\mathbf{B})$, onde **A** é sempre um intervalo contido em **B** por:

$$\int_{[a,b]} F = \cap_S \sum(F,S)$$

- **Integral de Rall**: Nesse método é dada a construção de uma integral intervalar onde, tanto o integrando como o intervalo de integração, são finitos. Caso os intervalos da integral tenham comprimentos iguais, não é necessário considerar todas as partições. Isso faz com que não seja necessário usar a soma de intervalos Riemann, que é o intervalo do intervalo (CAPRANI.; MADSEN; NIELSEN, 2002). É definida uma função intervalar Y em um intervalo $\mathbf{x} = [a,b]$, sendo $Y(x) = [\underline{y}(x), \bar{y}(x)]$, uma função que atribui um intervalo a cada $x \in \mathbf{x}$, onde $\underline{y}$ e $\bar{y}$ são denominadas como funções de extremidade de Y, as quais são arredondadas através da função $\triangledown Y(\mathbf{x})$:

$$\triangledown Y(\mathbf{x}) = [inf_x \in \mathbf{x}\{\underline{y}(x)\}, sup_x \in \mathbf{x}\{\bar{y}(x)\}]$$

  A integração de Rall é definida por:

$$\int_{\mathbf{x}} Y(x)\mathrm{d}x = \frac{w(\mathbf{x})}{n} \sum_{i=1}^{n} \triangledown Y(\mathbf{x}_i),$$

  onde $w(\mathbf{x})$ representa o tamanho do intervalo, o qual deve ser finito.

- **Integral de Bedregal**: Seja F uma função intervalar contínua sobre o intervalo **A** = [a,b]. Dada uma partição P de **A** definem-se os seguintes somatórios de Riemann:

$$\sigma(F,P) = \sum_{k=1}^{n} \prod F(\mathbf{A}_{[X_{k-1},x_k]}) dM(X_{k-1},x_k)$$

$$\sum(F,P) = \sum_{k=1}^{n} \coprod F(\mathbf{A}_{[X_{k-1},x_k]}) dM(X_{k-1},x_k),$$

  onde o primeiro representa a soma de Riemann inferior e o segundo a soma superior. A integral superior da função F de a até b é definida como:

$$\underline{\int_a^b} F(X)\mathrm{d}x = \coprod_{P \in \beta[a,b]} \sigma(F,P)$$

  e a integral inferior da mesma função como:

$$\overline{\int_a^b} F(X)\mathrm{d}x = \prod_{P \in \beta[a,b]} \sum(F, P)$$

Seja $F : I[a, b] \to \mathbb{I}(\mathbb{R})$ uma função intervalar contínua, então ela é uma função integrável (BEDREGAL; BEDREGAL, 2010) e a seguinte equação é validada:

$$\int_{\mathbf{A}}^{\mathbf{B}} F(\mathbf{X})dx = \left[\int_{\underline{a}}^{\underline{b}} F_l(x)dx, \int_{\underline{a}}^{\underline{b}} F_r(x)dx\right] \frac{dM(a,b)}{\underline{b}-\underline{a}}$$

- **Método de Simpson intervalar**: O método de Simpson intervalar (CAPRANI.; MADSEN; NIELSEN, 2002) é uma extensão do método de Simpson real (RUGGIERO; LOPES, 1996). Sendo uma função F quatro vezes continuamente derivável em A = [a, b] e as extensões intervalares das funções fef^{iv} são conhecidas, o método encontra um intervalo para a integral definida. O intervalo é dividido em n subintervalos igualmente espaçados, e, em cada um deles é aplicado o seguinte polinômio interpolador:

$$S_i = \frac{w(\mathbf{A}_i)}{6}\left(F(\underline{\mathbf{A}_i}) + 4f(m(\mathbf{A}_i)) + f(\bar{\mathbf{A}}_i)\right)$$

Dado um intervalo A e uma função f, a fórmula da integração numérica pelo método de Simpson intervalar é:

$$\int_{\underline{A}}^{\bar{A}} f(x)\mathrm{d}x = \sum_{i=1}^{n} \frac{w(\mathbf{A}_i)}{6}\left(F(\underline{\mathbf{A}_i}) + 4f(m(\mathbf{A}_i)) + f(\bar{\mathbf{A}}_i)\right) - \frac{w(\mathbf{A}_i)^5}{2880} G(\mathbf{A}_i)$$

Todos os métodos apresentados nessa seção serão utilizados na resolução da forma intervalar da função densidade de probabilidade com distribuição Weibull, a fim de comparar os resultados retornados por cada um dos métodos, para assim decidir qual deles melhor se adequa à função em questão.

2.3 IntPy

O pacote IntPy foi desenvolvido por Barreto para a linguagem de programação Python com o objetivo de implementar o tipo de variáveis intervalo e as operações sobre esse mesmo tipo.

O suporte para realização das operações aritméticas sobre intervalos foi feita usando um tipo especial de arredondamento, o arredondamento direcionado (KULISCH; MIRANKER, 1981).

Varjao (2011), com o uso do pacote IntPy, desenvolveu extensões intervalares para algumas funções, como, por exemplo, a logarítmica (log), potenciação (pow), entre outras. Todas as operações desenvolvidas foram feitas sobre o tipo *IReal* (Intervalo Real), o qual é implementado em IntPy através do uso de um subpacote.

Ex.: Para fazer a função potência sobre intervalos, usando IntPy, a chamada da função de potenciação (pow) seria:

$$pow([\underline{x}, \bar{x}], n),$$

onde $[\underline{x}, \bar{x}]$ é o intervalo que será elevado à potência n.

A implementação das operações em forma intervalar, como a apresentada acima, é feita através do critério proposto por Kulisch (1981), o semimorfismo. Levando em consideração a possibilidade do controle de erros numéricos poder ser feita utilizando intervalos no lugar de valores reais, ele propôs que a implementação da aritmética intervalar seja feita usando a aritmética de exatidão máxima, o que faz com que os resultados retornados sejam números de ponto flutuante ou um valor que esteja encapsulado em um intervalo cujos extremos são valores de ponto flutuante consecutivos.

2.4 Teoria da probabilidade

A teoria da probabilidade é o ramo da matemática que trata os fenômenos aleatórios, os quais podem ser repetidos sob a mesma condição, apresentam um conjunto de possíveis resultados e, após ser repetido diversas vezes, é possível visualizar um padrão.

Na abordagem de problemas práticos o uso de variáveis aleatórias é de grande utilidade, já que esse tipo de variável é usada para representar um conjunto de valores não-enumeráveis, o que frequentemente ocorre quando problemas práticos são tratados. Os modelos probabilísticos para variáveis aleatórias contínuas são chamados de funções densidade de probabilidade e podem envolver mais de um parâmetro (CAMPOS, 1997).

Para calcular probabilidades primeiramente é atribuído um peso de probabilidade $P(x)$ a todos os elementos do espaço amostral do fenômeno analisado, esse peso indica a probabilidade desse resultado específico ocorrer. O peso de probabilidade deve ser um valor positivo e a soma dos pesos de todos os elementos do espaço amostral deve totalizar 1. Em outras palavras, a probabilidade P de um evento E é definida como o peso dos elementos de E: $P(E) = \sum_{x:x \in E} P(x)$ (STEIN; DRYSDALE; BOGART, 2013).

A atribuição de um valor para cada elemento do espaço amostral é denominada como variável aleatória (ou variável estocástica) ou função aleatória (ou função estocástica). Caso a variável aleatória assuma ou um número finito ou um número infinito enumerável, ela é chamada de variável aleatória discreta, e, quando é usado um número infinito não-enumerável de valores ela é definida como uma variável aleatória contínua.

2.5 Métricas de qualidade de intervalo

Segundo Ratschek e Raknek (1988) os computadores usam aritmética de ponto flutuante em seus cálculos, onde os números reais são aproximados para um subconjunto dos reais, denominado representação numérica de máquina. Nesse processo de aproximação o número de máquina resultante apresenta alguma diferença com o valor exato e essa diferença é chamada de erro, o qual pode ser de dois tipos: o primeiro ocorre na aproximação do número real para número de máquina enquanto o segundo é originado pela aproximação de resultados intermediários (LORETO, 2006).

As técnicas intervalares fornecem uma ferramenta capaz de estimar e controlar automaticamente os erros onde, no lugar de aproximar um valor real x para um número de máquina, esse valor, normalmente desconhecido, é aproximado para um intervalo $\mathbf{x}$, o qual tem números de máquina como seus extremos inferior e superior, de maneira com que o valor x esteja contido nesse intervalo $\mathbf{x}$. Para medir a qualidade da aproximação feita no intervalo pode-se calcular o seu diâmetro. Os cálculos são realizados através do uso de intervalos, fazendo com que, consequentemente, seja necessário substituir a aritmética real e suas operações pela aritmética intervalar e as operações intervalares. Utilizando intervalos na computação é possível implementar estimativas para o erro em relação aos intervalos encontrados, são elas:

- Diâmetro: $\mathrm{w}(\mathbf{x}) = max(\mathbf{x}) - min(\mathbf{x})$;
- Erro absoluto: $|x - m(\mathbf{x})| < \frac{w(\mathbf{x})}{2}$, onde $m(\mathbf{x}) = \frac{\underline{\mathbf{x}}+\bar{\mathbf{x}}}{2}$ é o ponto médio do intervalo;
- Erro relativo: $|\frac{x-m(\mathbf{x})}{x}| \leq \frac{w(\mathbf{x})}{2min|\mathbf{x}|}$ se $0 \notin x$.

Nas medidas de erro vistas acima o ponto médio $\mathrm{m}(\mathbf{x})$ do intervalo $\mathbf{x}$ é utilizado para medir a distância do valor real em relação ao valor pontual do intervalo.

Essas medidas de erro são aplicadas nos intervalos obtidos nas funções densidade de probabilidade para, dessa maneira, medir a qualidade dos intervalos encontrados.

3 TRABALHOS RELACIONADOS

Esse capítulo trata dos trabalhos considerados relevantes e relacionados ao tema proposto por esse trabalho, mostrando-os sucintamente em cada seção, apresentando seus conceitos e resultados obtidos, os quais serão utilizados no desenvolvimento desse trabalho.

3.1 Extensão intervalar para as variáveis aleatórias com distribuições Uniforme, Normal, Gama, Exponencial e Pareto

O trabalho desenvolvido por Finger (2014) teve como objetivo definir com entradas intervalares funções densidade de probabilidade com distribuição de Pareto e Gama, analisando a qualidade do intervalo e o esforço computacional dos algoritmos implementados. São apresentados dois métodos para obter um intervalo solução para as funções densidade de probabilidade das variáveis aleatórias contínuas: usando o método de Simpson intervalar (CAPRANI.; MADSEN; NIELSEN, 2002) e usando a primitiva da função. Além disso, são mostrados os resultados da análise da qualidade dos intervalos encontrados em cada função densidade de probabilidade, tendo como métricas os erros absoluto e relativo, bem como o diâmetro do intervalo.

Com a análise da complexidade dos algoritmos implementados para cada função densidade de probabilidade, foram comparados os resultados usando o método de Simpson intervalar e a primitiva da função, a comparação é vista na Tabela 1.

Tabela 1 – Esforço computacional das distribuições

Distribuição	**Complexidade Real**		**Complexidade Intervalar**	
	Primitiva da Função	Simpson	Primitiva da Função	Simpson
Uniforme	O(1)	-	O(1)	-
Exponencial	O(1)	$O(2^n)$	-	$O(2^n)$
Normal	-	$O(2^n)$	-	$O(2^n)$
Gama	-	$O(2^n)$	-	$O(2^n)$
Pareto	O(1)	$O(2^n)$	O(1)	$O(2^n)$

Com os resultados, a autora conclui que, quando a primitiva da função é usada como solução os algoritmos são executados com menos esforço computacional, possuindo menor complexidade. A mesma conclusão também é formada em relação aos resultados numéricos nas formas reais e intervalares, onde as melhores soluções são obtidas através do uso da primitiva da função.

3.2 Estatística Descritiva Intervalar

Nsse trabalho, desenvolvido por Loreto (2006), foram definidos indicadores estatísticos com abordagem intervalar. Para isso, ela reuniu indicadores já definidos de forma intervalar como a média intervalar, moda intervalar, variância intervalar, desvio padrão intervalar, covariância intervalar e coeficiente de correlação intervalar e definiu, através do método de extensão intervalar, coeficiente de variação intervalar, mediana intervalar, quartil, decil e percentil intervalares.

A Figura 1 apresenta todas as expressões dos indicadores intervalares definidos no trabalho de Loreto.

Média Intervalar:	$\mathbf{ME}_v = [\underline{me}, \overline{me}] = \frac{1}{n}[\sum_{i=1}^{n} \underline{x}_i, \sum_{i=1}^{n} \overline{x}_i]$
Mediana Intervalar:	$\mathbf{MD}_v = [\underline{md}, \overline{md}] = \begin{cases} \frac{(\mathbf{x}_{(\frac{n}{2})} + \mathbf{x}_{(\frac{n}{2}+1)})}{2}, & \text{se } n \text{ for par} \\ (\mathbf{x}_{(\frac{n+1}{2})}), & \text{se } n \text{ for ímpar.} \end{cases}$
Moda Intervalar:	$\mathbf{MO}_v = [\underline{mo}, \overline{mo}] = [mo_{1 \leq i \leq n}\{\underline{x}_i\}, mo_{1 \leq i \leq n}\{\overline{x}_i\}]$
Amplitude total Intervalar:	$\mathbf{AT}_v = [\underline{at}, \overline{at}] = \begin{cases} [\underline{ma} - \overline{mi}, \overline{ma} - \underline{mi}], & \text{se } \underline{ma} \geq \overline{mi} \\ [0, \overline{ma} - \underline{mi}], & \text{se } \underline{ma} < \overline{mi}. \end{cases}$
Variância Intervalar:	$\mathbf{VA}_v = [\underline{va}, \overline{va}] = \frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{x_i} - \mathbf{ME})^2$
Desvio padrão Intervalar:	$\mathbf{DP}_v = [\underline{dp}, \overline{dp}] = \sqrt{\mathbf{VA}} = +\sqrt{[\underline{va}, \overline{va}]} = [+\sqrt{\underline{va}}, +\sqrt{\overline{va}}]$
Coeficiente de variação Intervalar:	$\mathbf{CV}_v = [\underline{cv}, \overline{cv}] = \frac{\mathbf{DP}}{\mathbf{ME}} = \frac{[\underline{dp}, \overline{dp}]}{[\underline{me}, \overline{me}]}$
Covariância Intervalar:	$\mathbf{CO}_v = [\underline{co}, \overline{co}] = \frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{x_i} - \mathbf{ME_X})(\mathbf{y_i} - \mathbf{ME_Y})$
Coeficiente de correlação Intervalar:	$\mathbf{CC}_v = [\underline{cc}, \overline{cc}] = \frac{\mathbf{CO}}{\mathbf{DP_X DP_Y}} = (\frac{[\underline{co}, \overline{co}]}{[\underline{dp}_X, \overline{dp}_X][\underline{dp}_Y, \overline{dp}_Y]})$
Separatrizes Intervalares:	$\mathbf{Q}_v = [\underline{q}, \overline{q}], \mathbf{D}_v = [\underline{d}, \overline{d}], \mathbf{P}_v = [\underline{p}, \overline{p}], pos = (n-1)\alpha + 1$

Figura 1 – Indicadores Estatísticos intervalares

Para comprovar a qualidade dos intervalos solução obtidos através do uso de extensão intervalar, ela comparou os resultados reais com os intervalares para cada um dos indicadores. A qualidade de aproximação dos intervalos solução foi verificada com o uso de cálculos das medidas de erro, onde, em todos os exemplos considerados no trabalho, certificou-se que a qualidade do intervalo foi mantida.

3.3 Interval enclosures for reliability metrics

Esse artigo de Mendonça e Campos (2016) foca na utilização de intervalos para limitar erros numéricos resultantes de processo de computar métricas de confiabilidade em máquinas digitais para as distribuições Exponencial, Weibull e Normal.

As métricas de confiabilidade são os aspectos levados em consideração na avaliação da confiabilidade (função da confiabilidade, função da taxa de problemas e tempo médio para falhas). Essas métricas são representas por números reais, logo, ao computar esses valores, o ambiente computacional gerará erros de aproximação, como os de arredondamento e truncamento. Para controlar esses erros numéricos esse trabalho implementou funções intervalares, as quais suportam o uso de intervalos nas distribuições de probabilidade para variáveis aleatórias contínuas.

Os intervalos encontrados pelas funções intervalares foram comparados com os resultados do software SHARPE (Symbolic Hierarchical Automated Reliability and Performance Evaluator), o qual é usado na área de confiabilidade e análise de performance (HIREL et al., 2011). Com os resultados do trabalho, definiu-se de forma intervalar as métricas de confiabilidade e as implementou nas distribuições de probabilidade propostas. Por fim, usando os valores reais computados pelo software SHARPE foi possível comprovar que os intervalos encontrados encapsulavam tais valores.

4 DEFINIÇÃO INTERVALAR DA FUNÇÃO DENSIDADE DE PROBABILIDADE COM DISTRIBUIÇÃO WEIBULL

Nesse capítulo será apresentada a definição intervalar da função densidade de probabilidade de uma variável aleatória contínua com distribuição Weibull. Tal definição foi desenvolvida usando o método de extensão intervalar com a aritmética intervalar de Moore (1966). Para comparar os resultados encontrados e a exatidão dos mesmos foi feita a implementação dos códigos da função densidade de probabilidade usando o pacote de extensão intervalar IntPy (BARRETO, 2016) da linguagem de programação Python, tanto para entradas com valores reais quanto para com entradas intervalares. Após a implementação foi realizada a análise numérica dos resultados obtidos, o que foi feito através da comparação dos resultados reais com os intervalares, utilizando as métricas de erros, para assim estimar a qualidade do intervalo solução encontrado. Os resultados dessas comparações podem ser vistos na seção 4.2. Na secão 4.3 será mostrada a análise de complexidade de cada um dos métodos testados na resolução da função com distribuição Weibull.

Existem dois métodos principais para criar a definição intervalar de uma uma função: o método da imagem intervalar e o método da extensão intervalar. Partindo dos trabalhos de indicadores estatísticos intervalares feitos por Loreto (2006) e extensão intervalar para variáveis aleatórias com distribuições Uniforme, Normal, Gama, Exponencial e Pareto, desenvolvido por Finger (2014), os quais obtiveram resultados satisfatórios usando o método de extensão intervalar, o mesmo método foi usado na definição intervalar da função densidade de probabilidade da distribuição Weibull nesse trabalho.

Para implementar a função, foi escolhida a linguagem de programação Python e o pacote IntPy. Essa escolha foi feita tendo como base o trabalho de Balboni et al. (2015), o qual apresenta os critérios para análise e escolha de ambientes intervalares. Os ambientes analisados foram os seguintes: Maple intervalar, IntLab, IntPy, C-XSC, Fortran-XSC, Pascal-XSC e Java-XSC. Os critérios levados em consideração na escolha foram questões relativas ao projeto de linguagem (legibilidade, manutenibilidade, simplicade, ortogonalidade, suporte a abstração, verificação de tipos e manipulação de exceções) e a qualidade do intervalo retornado pelo ambiente estudado. Com os resultados do trabalho, o ambiente que teve um melhor desempenho foi o pacote IntPy, tanto na qualidade do intervalo encontrado como no tempo de processamento. Além desses fatores o IntPy também é de fácil utilização e instalação, é bem documentado e apresenta resultados mais próximos do esperado.

Segundo Ratschek e Rokne (1988), os computadores usam aritmética de ponto flutuante, na qual números reais são aproximados para um conjunto numérico denominado representação numérica de máquina. Esse processo de aproximação gera dois tipos de erros, o primeiro consiste na aproximação feita quando o valor de entrada é aproximado para um número de máquina, e o segundo é causado pelos resultados intermediários aproximados por um número de máquina. O controle desses erros numéricos pode ser feito usando intervalos no lugar de números reais (KULISCH; MIRANKER, 1981). A implementação da aritmética intervalar seria feita usando a aritmética de exatidão máxima, onde os resultados seriam, ou um número de ponto flutuante ou um intervalo cujos limites inferior e superior são números de ponto flutuante consecutivos. Isso ocorre por que, quando é feita a sua implementação, a aritmética intervalar está definida sobre o sistema de ponto flutuante, e não dos números reais, $\mathbb{R}$ (BARRETO, 2016).

Na aritmética de exatidão máxima, os intervalos são definidos de modo que seus limites inferior e superior possuam arredondamento direcionado (KULISCH; MIRANKER, 1981), onde o intervalo solução de uma soma, por exemplo, seria definido da seguinte maneira:

$$\mathbf{X} + \mathbf{Y} = [\nabla(\underline{x}, \underline{y}), \triangle\ (\bar{x}, \bar{y})],$$

onde $\mathbf{x}$ e $\mathbf{y}$ são os intervalos somados, ∇ representa o arredondamento direcionado para baixo e $\triangle$ o arredondamento direcionado para cima.

4.1 Definicão intervalar

A distribuição de probabilidade de uma variável aleatória contínua X é uma descrição do conjunto de probabilidades associadas com todos os valores possíveis para essa variável. E uma função densidade de probabilidade pode ser usada para descrever a distribuição de probabilidades de uma variável aleatória contínua X, onde a probabilidade dessa variável estar entre dois valores a e b é dada por uma integral com limites a e b. Uma função densidade de probabilidade pode ser vista na equação a seguir:

$$P(a < X < b) = \int_a^b .$$

Uma variável é definida como uma variável aleatória contínua caso exista uma função $f_x : \mathbb{R} \to [0, \infty)$ que obedeça às seguintes propriedades:

- $f_x(x) \geq 0, \forall x \in \mathbb{R}$,
- $\int_{-\infty}^{\infty} f(x)dx = 1$.

Dessa maneira a variável aleatória contínua pode assumir uma quantidade não-enumerável e f(x) é chamada de função densidade de probabilidade dessa variável.

A distribuição cuja função densidade de probabilidade será definida com entradas intervalares será a distribuição de Weibull, a qual é uma distribuição de probabilidade para variáveis aleatórias contínuas que é aplicada em diversos casos devido a sua versatilidade e relativa simplicidade (NAVIDI, 2016). A função densidade de probabilidade da distribuição Weibull tem dois parâmetros de entrada, os quais definem sua forma e escala. A função densidade de probabilidade com essa distribuição é vista na seguinte equação

$$f(x) = \int_a^b \alpha\beta x^{\alpha-1} e^{-(\beta x)^\alpha} = e^{-(\beta b)^\alpha} - e^{-(\beta a)^\alpha}, x > 0, \tag{4.1}$$

onde α representa o parâmetro de forma e β o parâmetro de escala da função

Se x é uma variável aleatória contínua com parâmetros α e β, sua notação é $x\ Weibull(\alpha, \beta)$. Quando $\alpha = 1$, a distribuição de Weibull assume a mesma forma que a distribuição exponencial com parâmetros $\alpha = \beta$ (NAVIDI, 2016).

A figura 4.1 mostra alguns exemplos com valores variados dos parâmetros de entrada. Com a variação de α e β podem ser geradas diversas formas de curva, o que permite que a distribuição de Weibull possa ser usada em uma vasta gama de casos, o que é o principal ponto positivo dessa distribuição, a sua versatilidade (NAVIDI, 2016).

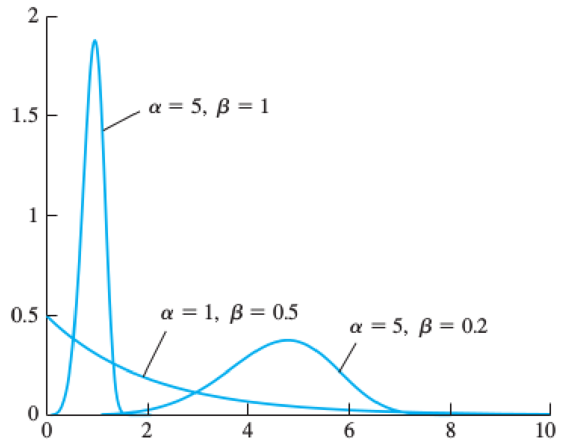

Figura 2 – Exemplos de curvas da distribuição Weibull.

Fonte:(NAVIDI, 2016)

A definição intervalar da função densidade de probabilidade de uma variável aleatória contínua com distribuição Weibull, usando o método de extensão intervalar é expressa na equação a seguir:

$$Weibull(\mathbf{X}, \alpha, \beta) = \int_a^b \alpha\beta^\alpha \mathbf{X}^{\alpha-1} e^{-(\beta\mathbf{X})^\alpha} = e^{-(\beta b)^\alpha} - e^{-(\beta a)^\alpha} \tag{4.2}$$

onde $\mathbf{X}$ representa o intervalo. Nessa equação todas as entradas reais da função apresentada na equação 4.1 foram trocadas por entradas intervalares que contenham o valor real.

Após definida a função com entradas intervalares, o próximo passo é realizar a implementação com aplicação de diferentes métodos de integração

4.2 Análise numérica

Na resolução da integral da função densidade de probabilidade, será escolhido, dentre os vários métodos para integrais intervalares apresentados na seção 2.2, o que melhor atende aos objetivos do trabalho. Essa escolha será feita através da implementação de cada um desses métodos e analisando a qualidade do intervalo solução e a complexidade dos algoritmos implementados.

Para obter resultados numéricos foram utilizados 3 conjuntos de dados de entrada como parâmetros tanto para a função com entrada real quanto para a função com entrada intervalar. Tais conjuntos de valores foram retirados de exemplos encontrados na literatura e foram utilizados devido ao fato de que seus resultados reais já são conhecidos, fazendo com que os resultados intervalares obtidos possam ter sua validade confirmada. Os conjuntos de valores foram então aplicados na integral da função densidade de probabilidade com distribuição Weibull, sendo eles usados na resolução da função.

- Ex. 1: Intervalo $\mathbf{x} = [2, 7]$, $\alpha = 0.3, \beta = 0.1$
- Ex. 2: Intervalo $\mathbf{x} = [2, 4]$, $\alpha = 0.5, \beta = 3.0$
- Ex. 3: Intervalo $\mathbf{x} = [1, 2]$, $\alpha = 0.75, \beta = 1.0$

Como a distribuição precisa de resolução de uma integral, foram aplicados alguns métodos de integração intervalar encontrados na literatura, além da primitiva da função, a fim de comparar qual o melhor método de resolução para a função com distribuição de Weibull.

A tabela 2 mostra os resultados obtidos na implementação da função com distribuição Weibull, mostrando os intervalos solução retornados por cada um dos métodos de resolução de integrais intervalares testados.

Tabela 2 – Resultados usando exemplos e exercícios propostos por Navidi (2016).

Resolução	Ex.1	Ex. 2	Ex.3
Prim. real	0.1323719964625507	0.05503651641542917	0.18183930273552706
Simpson	0.13237199646257727	0.0550365164154236	0.18183930273551693
Primitiva Intervalar	[0.1323719964625506, 0.1323719964625507]	[0.05503651641542917, 0.05503651641542917]	[0.18183930273552706, 0.18183930273552706]
Bedregal Intervalar	[0.1323551818384106, 0.13238881428789467]	[0.055028797091662585, 0.05504423671664428]	[0.1818237349866751, 0.18185487168747985]
Moore Intervalar	[0.1323551818384106, 0.13238881428789467]	[0.055028797091662585, 0.05504423671664428]	[0.1818237349866751, 0.18185487168747985]
Rall Intervalar	[0.13235518183841055, 0.1323888142878943]	[0.05502879709166212, 0.055044236716645004]	[0.18182373498667925, 0.1818548716874769]
Simpson Intervalar	[0.1323719964624741, 0.13237199646259587]	[0.05503651641540019, 0.05503651641545183]	[0.18183930273551388, 0.1818393027356018]

Analisando os resultados mostrados na tabela 2 é possível perceber que o valor real de todos os exemplos está contido nos intervalos solução encontrados por cada um dos métodos de integração intervalar testados.

A fim de adicionar no critério de escolha do método o tempo de processamento, a tabela 3 mostra os tempos de execução dos métodos para integrais intervalares, onde o mesmo é calculado através de uma média do tempo obtido em 10 execuções do cálculo da função.

Tabela 3 – Tempo de execução dos exemplos para cada método testado.

Tempo(s)	Ex. 1	Ex. 2	Ex. 3
Prim. real	1.78813934326e-05	1.19209289551e-05	2.21729278564e-05
Simpson 1/3	0.00646996498108	0.00646185874939	0.00589609146118
Prim. Intervalar	0.000240087509155	0.000144004821777	0.000274896621704
Bedregal Intervalar	0.502429962158	0.478626012802	0.490545034409
Moore Intervalar	0.469722986221	0.468303203583	0.505259990692
Rall	0.411931991577	0.447436094284	0.420495986938
Simpson Intervalar	0.350502967834	0.351670026779	0.345978975296

Com os tempos mostrados na tabela 3, pode-se ver, como esperado, que os menores tempos de execução foram vistos na resolução da primitiva com entradas reais da função, seguida pela primitiva com entradas intervalares. Já, dentre os métodos de integração intervalar, o que levou menos tempo para concluir sua execução foi o método de Simpson, que obteve o menor tempo nos 3 exemplos testados.

A fim de comprovar a qualidade dos resultados obtidos, os intervalos foram alvo de análise de erros, onde foram utilizadas as métricas de qualidade de intervalos, as quais são: o erro absoluto, o qual consiste na diferença entre o valor exato de um número e a aproximação; o erro relativo, o qual pode ser calculado pelo quociente entre o valor absoluto e o aproximado; e o diâmetro de intervalo, que é a diferença entre os limites superior e inferior do intervalo.

Para todos os resultados encontrados foi utilizado o mesmo computador com as seguintes configurações: processador Intel Core i3-4005U @1.70 GHZ x2, Memória RAM de 3.8GB, sistema operacional Linux Mint 17.3 Cinnamon 64-bit.

Todos os resultados apresentados, tanto reais quanto intervalares, utilizaram o sistema de ponto flutuante F(10, 15, -18, 18).

A tabela 4 mostra os erros absoluto e relativo e o diâmetro dos resultados retornados pelo cálculo da primitiva da função com entradas intervalares e da integral da função com os métodos de integração intervalar usando como valores de entrada os parâmetros do exemplo 1.

Tabela 4 – Erros absoluto, relativo e diâmetro do exemplo 1.

Ex. 1	Erro absoluto	Erro relativo	Diâmetro
Prim. Interv.	5.551115123125783e-17 < 5.551115123125783e-17	4.193572108505778 7e-16≤ 7.348121914416173e-18	1.110223024625156e-16
Bedregal	1.6006019287750917e-9 < 1.681622474203337e-5	1.2091695914157471e-8 ≤ 2.225714483567406e-6	3.363244948406674e-5
Moore	1.6006019287750917e-9 < 1.681622474203337e-5	2.2384029843571294e-11≤ 9.447462008336371e-8	3.363244948406674e-5
Rall	1.6006017067304867e-9 < 1.6816224741880714e-5	3.2324892526784245e-11≤ 1.1336939377474662e-7	3.363244948376143e-5
Simpson	4.278910559207816e-12 < 8.564509889480432e-7	3.2324892526784245e-11≤ 1.1336939377474662e-7	6.203371150093062e-14

Com os valores mostrados na tabela 4 pode-se notar que a diferença entre o valor exato e o ponto médio do intervalo solução encontrado pela forma intervalar da primitiva da função foi nulo, e os erros absoluto de todos os outros métodos testados obedeceram à condição do erro. Em relação ao erro relativo, todos os métodos testados respeitaram à função do erro, sendo a forma intervalar da primitiva da função a que obteve o menor erro em comparação com os outros métodos implementados. No cálculo do diâmetro dos intervalos, o método que obteve o menor intervalo foi a forma intervalar da primitiva da função, seguida pelo método de Simpson intervalar.

A tabela 5 mostra os erros absoluto e relativo e o diâmetro dos resultados retornados pelo cálculo da primitiva da função com entradas intervalares e da integral da função com os métodos de integração intervalar usando como valores de entrada os parâmetros do exemplo 2.

Tabela 5 – Erros absoluto, relativo e diâmetro do exemplo 2.

Ex. 2	Erro absoluto	Erro relativo	Diâmetro
Prim. Interv.	—	0.0 ≤ 0.0	0.0
Bedregal	4.887242624485033e-10< 7.719812490848688e-6	8.879999939667189e-9≤ 4.2481199514459476e-7	1.5439624981697375e-5
Moore	4.887242624485033e-10< 7.719812490848688e-6	1.2658357254800408e-11≤ 1.80321679412831e-8	1.5439624981697375e-5
Rall	4.887243942874875e-10< 7.719812491441963e-6	1.9695148655903736e-11≤ 2.1638570997428777e-8	1.5439624982883926e-5
Simpson	1.0839523723049638e-12< 3.9317033470251395e-7	1.9695148655903736e-11≤ 2.1638570997428777e-8	2.631228568361621e-14

Com os valores mostrados na tabela 5 pode-se notar que a diferença entre o valor exato e o ponto médio do intervalo solução encontrado pela forma intervalar da primitiva da função foi nulo, devido aos intervalos solução serem degenerados (onde os limites do intervalo são iguais), e os erros absoluto de todos os outros métodos testados obedeceram à condição do erro. Em relação ao erro relativo, todos os métodos testados respeitaram à condição do erro, sendo a forma intervalar da primitiva da função a qual obteve o menor erro, onde o valor desse erro foi nulo em comparação com os outros métodos implementados. No cálculo do diâmetro dos intervalos, o método que obteve o menor intervalo foi a forma intervalar da primitiva da função, seguida pelo método de Simpson intervalar.

A tabela 6 mostra os erros absoluto e relativo e o diâmetro dos resultados retornados pelo cálculo da primitiva da função com entradas intervalares e da integral da função com os métodos de integração intervalar usando como valores de entrada os parâmetros do exemplo 3.

Tabela 6 – Erros absoluto, relativo e diâmetro do exemplo 3.

Ex. 3	Erro absoluto	Erro relativo	Diâmetro
Prim. Interv.	—	$0.0 \leq 0.0$	0.0
Bedregal	6.015504205869604e-10 < 1.5568350402381248e-5	3.308143022643871e-9 $\leq$ 2.8306956177422647e-6	3.1136700804762496e-5
Moore	6.015504205869604e-10 < 1.5568350402381248e-5	3.6471308115628045e-12$\leq$ 2.8306956177422647e-6	3.1136700804762496e-5
Rall	6.015510312096239e-10< 1.5568350398828534e-5	3.308146380678434e-9 $\leq$ 2.830695617096362e-6	3.113670079765707e-5
Simpson	1.1369516439430072e-12< 7.928969183168544e-7	6.2525077188435236e-12$\leq$ 1.4417919408326336e-7	8.79296635503124e-14

Com os valores mostrados na tabela 6 pode-se notar resultados semelhantes aos mostrados na tabela 5. A diferença entre o valor exato e o ponto médio do intervalo solução encontrado pela forma intervalar da primitiva da função foi nulo, devido ao intervalo solução ser degenerado, e os erros absoluto de todos os outros métodos testados obedeceram à condição do erro. Em relação ao erro relativo, todos os métodos testados respeitaram à condição do erro, sendo a forma intervalar da primitiva da função a qual obteve o menor erro, onde o valor desse erro foi nulo. comparação com os outros métodos implementados. No cálculo do diâmetro dos intervalos, o método que obteve o menor intervalo foi a forma intervalar da primitiva da função, seguida pelo método de Simpson intervalar.

4.3 Análise de complexidade

Um dos critérios levados em consideração na comparação dos métodos de integracão intervalar foi a complexidade dos algoritmos que implementavam cada um dos métodos. No contexto de algoritmos o termo complexidade se refere à quantidade de trabalho despendido pelo programa na resolução de um dado problema (TOSCANI; VELOSO, 2001).

A complexidade de um algoritmo em termos de espaço de computação e espaço de memória são fatores importantes para a definição da eficiência de um algoritmo (TOSCANI; VELOSO, 2001). Quando um algoritmo obtém a solução para um problema em tempo polinomial, ele é tido como razoável (KREINOVICH et al., 1998). Caso exista um algoritmo de tempo polinomial que resolva todas as instâncias de um problema, então este problema é definido como tratável e como intratável em caso contrário (TOSCANI; VELOSO, 2001).

Nessa seção é apresentada a análise de complexidade feita para cada um dos métodos utilizados na resolução da função com distribuições, tanto para os métodos que utilizavam a primitiva da função, quanto para os métodos de integração intervalar comparados nesse trabalho.

- Primitiva intervalar: Para realizar o cálculo da primitiva da função densidade de probabilidade com distribuição Weibull é necessário passar como parâmetros de entrada α e β, além do intervalo onde se deseja calcular a probabilidade. Após isso, todas as operações aritméticas reais foram substituídas por operações intervalares correspondentes. Com isso pode-se definir a complexidade desse algoritmo como de ordem constante, O(1).

- Método de Simpson intervalar: A forma intervalar do método de Simpson realiza operações aritméticas intervalares. Nesse método é criada uma lista onde são armazenadas todas as n subdivisões do intervalo, o que faz com que seja necessário o uso de um laço com n execuções Após isso é executado outro laço, onde o método é calculado sobre as n subdivisões. Devido a isso, a complexidade computacional desse método não depende dos parâmetros de entrada, mas sim das subdivisões de método. Como a eficiência desse método depende do número de subdivisões, a complexidade do mesmo é de ordem $O(2^n)$.

- Integral de Bedregal: Esse método de resolução de integrais intervalares realiza operações aritméticas intervalares. Nesse método é criada uma lista onde são armazenadas todas as n subdivisões do intervalo, o que faz com que seja necessário o uso de um laço com n execuções. Após isso é executado outro laço, onde o método é calculado sobre as n subdivisões. Devido a isso, a complexidade computacional desse método não depende dos parâmetros de entrada, mas sim das subdivisões de método. Mesmo que o método em questão vise integrar sobre dois intervalos, diferentemente dos outros métodos, a eficiência do mesmo ainda depende do número de subdivisões, sendo sua complexidade de ordem $O(2^n)$.

- Integral de Moore: Esse método de resolução de integrais intervalares realiza operações aritméticas intervalares, o qual propõe um método mais geral para a integração intervalar. Nesse método é criada uma lista onde são armazenadas todas as n subdivisões do intervalo, o que faz com que seja necessário o uso de um laço com n execuções Após isso é executado outro laço, onde o método é calculado sobre as n subdivisões. Devido a isso, a complexidade computacional desse método não depende dos parâmetros de entrada, mas sim das subdivisões de método. Como a eficiência desse método depende do número de subdivisões, a complexidade do mesmo é de ordem $O(2^n)$.

- Integral de Rall: Esse método de resolução de integrais intervalares realiza operações aritméticas intervalares, nele são feitas simplificações sobre o método de integração de Moore, eliminando o uso de todas as partições do intervalo, sendo possível obter o resultado através de uma amostra do intervalo. Nesse método é criada uma lista onde são armazenadas todas as n subdivisões do intervalo, o que faz com que seja necessário o uso de um laço com n execuções Após isso é executado outro laço, onde o método é calculado sobre as n subdivisões. Devido a isso, a complexidade computacional desse método não depende dos parâmetros de entrada, mas sim das subdivisões de método. Como a eficiência desse método depende do número de subdivisões, a complexidade do mesmo é de ordem $O(2^n)$.

Na tabela 7 é apresentado um resumo das complexidades analisadas anteriormente.

Tabela 7 – Complexidade dos métodos usados no cálculo da forma intervalar da função

Métodos	Ordem de Complexidade
Primitiva Intervalar	O(1)
Integral de Bedregal	O(2^n)
Integral de Moore	O(2^n)
Integral de Rall	O(2^n)
Simpson Intervalar	O(2^n)

A partir da análise da complexidade dos algoritmos apresentados, pode-se perceber que, quando é utilizada a primitiva da função como solução, o algoritmo é executado com uma complexidade menor quando comparado aos métodos de integração, os quais possuem tempo de execução exponencial. O que faz com que o uso da primitiva da função ofereça menos trabalho na execução do cálculo.

A complexidade exponencial dos métodos de integração intervalar deve-se ao fato de que em todos eles é necessário subdividir o intervalo em n subintervalos e depois aplicar a função em cada um deles. Porém, como o mesmo acontece com a resolução da integral definida da função com entradas reais, o uso de métodos intervalares não causa um maior impacto no esforço computacional.

5 CONSIDERAÇÕES FINAIS

Quando se trabalha com computação numérica, um dos fatores mais importantes é a exatidão das respostas dos cálculos. Esse problema é recorrente no estudo de variáveis aleatórias contínuas no cálculo da função densidade de probabilidade de uma distribuição, já que o resultado dessa função, quando oriundo da computação numérica, é sujeito a erros de aproximação, como os de arredondamento e truncamento, devido à representação aritmética de ponto flutuante, a qual o computador, com seu conjunto finito de números representáveis, é incapaz de representar. Devido a esses problemas a matemática intervalar é utilizada, a qual fornece meios para controlar os erros de forma automática, assim retornando resultados mais exatos.

O estudo das variáveis aleatórias sobre o conjunto dos números reais $\mathbb{R}$ é uma das áreas onde a matemática intervalar pode ser aplicada devido aos problemas encontrados no cálculo das probabilidades desse tipo de variável, já que é necessária a resolução de uma integral definida da função densidade, a qual, na maioria das vezes não possui primitiva simples ou fácil de se obter. No caso da inexistência da primitiva da integral na forma analítica, como acontece nos casos das funções densidade das distribuições Normal e Gama, é utilizada a integração numérica para o cálculo da integral da função, o que faz com que possam ser gerados erros, como os de arredondamento e de truncamento, os quais podem ser controlados através do uso da matemática intervalar.

Nesse trabalho foi definida de forma intervalar a função densidade de probabilidade de uma variável aleatória com distribuição Weibull, o que foi feito através do uso da extensão intervalar, onde os valores de entrada reais foram substituídos por valores intervalares correspondentes. Após isso, foi feita a implementação da forma intervalar da função, para assim verificar qual dos métodos para resolução da função retornava os melhores resultados de maneira mais eficiente.

Para poder computar a forma intervalar da função densidade com distribuição Weibull, foi utilizada a linguagem de programação Python junto com o pacote de extensão intervalar IntPy. Com isso foi possível comparar os resultados de diferentes métodos para a resolução da função, a fim de verificar qual deles obteve melhores resultados de maneira mais eficiente.

A análise numérica dos resultados obtidos pelos métodos de integracão intervalar e pela primitiva da função com valores intervalares, foi feita no sistema de ponto flutuante (10,15,-18,18). Nessa análise verificou-se que todos os métodos implementados para resolver a forma intervalar da função densidade com distribuição Weibull retornavam intervalos solução que continham a probabilidade para valores reais.

Além da análise numérica, também foi feita a análise de complexidade dos métodos, a fim de comparar qual deles realiza o cálculo da forma intervalar com menos esforço computacional. Dos métodos comparados, o que mostrou maior eficiência foi a primitiva da função com valores intervalares, a qual realiza o cálculo em tempo polinomial

($O(1)$), enquanto os métodos de integração intervalar mostraram ser de ordem exponencial ($O(2^n)$), devido à necessidade de dividir o intervalo de integração em n subintervalos e depois aplicar a função em cada um desses subintervalos.

A partir dos resultados colhidos em ambas análises numérica e de complexidade, foi possível definir que a forma intervalar da primitiva da função com distribuição Weibull foi o método que retornou os melhores resultados e com maior eficiência. Dentre os métodos de integracão intervalar, com os conjuntos de valores testados, o que mostrou melhores resultados foi o método de Simpson intervalar, o que foi visto em todos os casos calculados.

O objetivo principal deste trabalho foi ressaltar a importância do uso da matemática intervalar na busca de uma maior exatidão nos cálculos de função densidades de probabilidade de uma variável aleatória contínua com distribuição Weibull e escolher o melhor método para calcular a forma intervalar da função.

Em trabalhos futuros pretende-se definir de forma intervalar as outras fórmulas presentes na distribuição Weibull, como as fórmulas da função de confiabilidade, função de risco, tempo médio de vida, variância de distribuição e função geradora de momentos. Além disso também deseja-se definir a forma intervalar da função densidade de probabilidade de uma variável aleatória com distribuição Weibull com 3 parâmetros, e, assim como no presente trabalho, também deverão ser comparados diversos métodos de integracão intervalar a fim de decidir qual é mais indicado a essa forma da funcão com distribuição Weibull.

REFERÊNCIAS

BALBONI, M. D. C. et al. Exatidão máxima no cálculo da probabilidade com distribuição normal da pressão arterial diastólica. In: **Proceeding Series of the Brazilian Society of Applied and Computational Mathematics**. São Carlos, Brasil: [s.n.], 2015. v. 3, n. 1. Citado na página 37.

BARRETO, R. M. Intpy 0.1.3. 2016. **Disponível em: <https://pypi.python.org/pypi/ IntPy/0.1.3>**, 2016. Citado 4 vezes nas páginas 22, 23, 37 e 38.

BEDREGAL, B. R.; BEDREGAL, R. C. A generalization of the moore and yang integral approach. **Computing**, 2010. Citado na página 29.

CAMPOS, M. A. **Uma Extensão Intervalar para a Probabilidade Real**. Tese (Doutorado) — Universidade Federal de Pernambuco, Recife, 1997. Citado 3 vezes nas páginas 21, 25 e 30.

CAPRANI., O.; MADSEN, K.; NIELSEN, H. Introduction to interval analysis. **IMM - Informatics and Mathematical Modelling**, 2002. Citado 3 vezes nas páginas 28, 29 e 33.

FERSON, S. et al. **Exact bounds on finite populations of interval data**. [S.l.]: Reliable Computing, 2004. 207-233 p. Citado na página 26.

FINGER, A. F. Extensão intervalar para as variáveis aleatórias com distribuições uniforme, normal, gama, exponencial e pareto. dissertação (mestrado). **Universidade Federal de Pelotas**, 2014. Citado 3 vezes nas páginas 21, 33 e 37.

HICKEY, Q. J. T.; ENDEN, M. H. van. Interval arithmetic: from principles to implementation. **Journal of the ACM**, 1999. Citado na página 26.

HIREL, C. et al. Reliability and performability modeling using sharpe 2000. 2011. Citado na página 34.

KEARFOTT, R. B. Interval computations: Introduction, uses, and resources. **Euromath Bulletin**, v. 2, p. 95–112, 1996. Citado na página 21.

KREINOVICH, V. Probabilities. intervals, what next? optimization problems related to extension of interval computations to situation with partial information about probabilities. **Global Optimization**, v. 29, n. 3, p. 265–280, 2003. Citado na página 26.

KREINOVICH, V. et al. **Computational Complexity and Feasibility of Data Processing and Interval Computations**. [S.l.]: Dordrecht: Kluwer, 1998. Citado na página 44.

KULISCH, U.; MIRANKER, L. **Computer Arithmetic in Theory and Practice**. 1th. ed. [S.l.]: Academic Press, 1981. Citado 3 vezes nas páginas 29, 30 e 38.

LORETO, A. B. **Análise da Complexidade Computacional de Problemas de Estatística Descritiva com Entradas Intervalares**. Tese (Doutorado) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2006. Citado 4 vezes nas páginas 21, 31, 33 e 37.

MENDONçA, A.; CAMPOS, M. A. Interval enclosures for reliability metrics. **Tendências em Matemática Aplicada e Computacional, 17, N. 2**, 2016. Citado na página 34.

MESQUITA, M. Matemática intervalar: princípios e ferramenta c-xsc. **Online**, 2004. Citado na página 21.

MOORE, R.; KEARFOTT, M.; CLOUD, J. **Introduction to Interval Analysis**. Philadelphia: Studies in Applied and Numerical Mathematics (SIAM), 2009. 213 p. Citado na página 25.

MOORE, R. E. **Interval Analysis**. Englewood Cliffs, NJ: Prentice Hall, 1966. 159 p. Citado 3 vezes nas páginas 25, 26 e 37.

MOORE, R. E. **On computing the range of a rational function of variables over a bounded region**. [S.l.]: Computing, 1976. 1-15 p. Citado 2 vezes nas páginas 21 e 26.

MOORE, R. E. **Methods and Applications of Interval Analysis**. 2. ed. Madison, Wisconsin: Studies in Applied and Numerical Mathematics (SIAM), 1979. 200 p. Citado na página 25.

MOORE., R. E. R. E.; YANG, C. T.; STROTHER, W. Interval integrals. **Technical Memorandum: Mathematics**, 1960. Citado na página 27.

NAVIDI, W. **Statistics for Engineers and Scientists**. [S.l.]: Mc Graw Hill Education, 2016. Citado 5 vezes nas páginas 17, 22, 38, 39 e 40.

NOBREGA, G. Integrais de linha intervalar: Fundamentos e aplicações. **Universidade Federal do Rio Grande do Norte**, 2010. Citado na página 27.

OLIVEIRA, P.; DIVERIO, T.; CLAUDIO, D. **Fundamentos de Matemática Intervalar**. Porto Alegre, Brasil: Sagra-Luzzato, 1997. 93 p. Citado na página 26.

RATSCHEK, H.; ROKNE, J. **New Computer Methods for Global Optimization**. Chichester, United Kingdom: Limited, 1988. 229 p. Citado 3 vezes nas páginas 21, 30 e 37.

RUGGIERO, M. A. G.; LOPES, V. L. R. **Cálculo numérico: aspectos teóricos e computacionais**. 2. ed. São Paulo: Pearson Makron Books, 1996. Citado na página 29.

SANTIAGO, R. H. N.; BEDREGAL, B. R. C.; ACIÓLY, B. M. Formal aspects of correctness and optimality of interval computations. **Formal Aspects of Computing**, Springer-Verlag, v. 18, p. 231–243, 2006. Citado na página 25.

STEIN, C.; DRYSDALE, R. L.; BOGART, K. **Matemática Discreta para Ciência da Computação**. [S.l.]: Pearson, 2013. Citado na página 30.

TOSCANI, L.; VELOSO, P. **Complexidade de Algoritmos: análise, projetos e métodos**. [S.l.]: Sagra-Luzzato, 2001. Citado 2 vezes nas páginas 43 e 44.

VARJÃO, F. R. G. **IntPy: Computação Científica Auto Validável em Python**. Dissertação (Mestrado) — Universidade Federal de Pernambuco, Recife, 2011. Citado na página 29.

Anexos

ANEXO A – IMPLEMENTAÇÃO DA FUNÇÃO DENSIDADE DE PROBABILIDADE DE VARIÁVEIS ALEATÓRIAS COM DISTRIBUIÇÃO WEIBULL INTERVALAR

```python
#! / u s r / b i n / env python
#coding: utf-8
from scipy.integrate import quad
from scipy import inf, integrate
import fpconst
from math import exp, expm1
from intpy import *
from intpy.support.stdfunc import sqrt
import math
from decimal import *
import time
from decimal import Decimal
import decimal
import time

def powI(dado, exp):
        if dado.inf == 0:
                result = IReal(0)
        else:
                result = IReal(dado.inf**exp.inf, dado.sup**exp.sup)
        return result

def weibullR(alfa, beta, a, b):
        begin_time = time.time()
        e=2.7182818284590452353602874713526624977572470936999595
        if a == 0:
                result = e ** -((beta * b) ** alfa)
        else:
                result = (1 - e ** -((beta * b) ** alfa)) - (1 - e ** -((beta * a) ** alfa))
        end_time = time.time()
        print(end_time - begin_time)
        return result

def weibullRSimpson(alfa,beta,a,b):
        begin_time = time.time()
        n = 5000
```

```
    total = 0
    e=2.7182818284590452353602874713526624977572470936999595
    tamanho = (a - b)/float(n)
    total = total+(alfa * (beta ** alfa) * a ** (alfa - 1) * e ** (-((beta * a) ** alfa)))
    total = total + (alfa * (beta ** alfa) * b ** (alfa - 1) * e ** (-((beta * b) ** alfa)))
    aux = b
    for i in range(n-1):
        if i%2 == 0:
            aux = aux + tamanho
            total = total + 4*(alfa * (beta ** alfa) * aux ** (alfa - 1) * e ** (-((beta * aux) ** alfa)))
        else:
            aux = aux + tamanho
            total = total + 2 * (alfa * (beta ** alfa) * aux ** (alfa - 1) * e ** (-((beta * aux) ** alfa)))
    total = (tamanho/3.0) * total

    print(end_time - begin_time)
    return -total

def weibullI(alfa, beta, a, b):
    begin_time = time.time()
    e=IReal(2.718281828459045235360287471352662497757247093699959)
    alfa = IReal(alfa)
    beta = IReal(beta)
    aI = IReal(a)
    bI = IReal(b)
    if a == 0:
        y = -(powI(beta * bI, alfa))
        result = powI(e, y)
    else:
        v = -(powI(beta * aI, alfa))
        w = -(powI(beta * bI, alfa))
        result = (IReal(1) - powI(e, w)) - (IReal(1) - powI(e, v))
    end_time = time.time()
    print(end_time - begin_time)
```

```
        return result

def weibullIRall(alfa, beta, a, b):
        begin_time = time.time()
        listaInt = []
        e=IReal
           (2.718281828459045235360287471352662497757247093699959)
        x = IReal(a, b)
        n = 5093
        total = IReal(0)
        alfa = IReal(alfa)
        beta = IReal(beta)
        h = (x.sup - x.inf)/float(n)
        aux = x.inf
        for i in range (n):
                listaInt += [IReal(aux, aux + h)]
                aux = aux + h
        for i in range (n):
                powX = powI(listaInt[i], alfa - IReal(1))
                powy = powI(beta * listaInt[i], alfa)
                powe = powI(e, -(powy))
                total = total + (alfa * (powI(beta, alfa))* powX
                   * powe)
        end_time = time.time()
        print(end_time - begin_time)
        return total * h

def weibullIBedregal(alfa, beta, a, b):
        begin_time = time.time()
        listaInt = []
        n = 5093
        e=IReal
           (2.718281828459045235360287471352662497757247093699959)
        total = IReal(0)
        x = IReal(a)
        y = IReal(b)
        alfa = IReal(alfa)
        beta = IReal(beta)
        h = (y.inf - x.inf)/float(n)
        aux = x.inf

        for i in range (n):
```

```
                listaInt += [IReal(aux, aux + h)]
                aux = aux + h

        for i in range (n):
                powX = powI(listaInt[i], alfa - IReal(1))
                powy = powI(beta * listaInt[i], alfa)
                powe = powI(e, -(powy))
                total = total + (alfa * (powI(beta, alfa))* powX * powe) * h

        limite_inf = abs(y.inf - x.inf)
        limite_sup = abs(y.sup - x.sup)

        if(limite_inf > limite_sup):
                dM = limite_inf
        else:
                dM = limite_sup

        bedrego = dM/(y.inf - x.inf)
        end_time = time.time()
        print(end_time - begin_time)
        return total * bedrego

def weibullIMoore(alfa, beta, a, b):
        begin_time = time.time()
        listaInt = []
        n = 5093
        e=IReal(2.7182818284590452353602874713526624977572470936999595)
        total = IReal(0)
        x = IReal(a)
        y = IReal(b)
        alfa = IReal(alfa)
        beta = IReal(beta)
        h = (y.inf - x.inf)/float(n)
        aux = x.inf
        for i in range (n):
                listaInt += [IReal(aux, aux + h)]
                aux = aux + h
        for i in range (n):
                powX = powI(listaInt[i], alfa - IReal(1))
                powy = powI(beta * listaInt[i], alfa)
```

```
                powe = powI(e, -(powy))
                total = total + (alfa * (powI(beta, alfa)) * powX
                    * powe) * h
        end_time = time.time()
        print(end_time - begin_time)
        return total

def weibullISimpson(alfa,beta, a, b):
        begin_time = time.time()
        listaInt = []
        n = 5092
        e=2.7182818284590452353602874713526624977572470936999595
        total = 0
        x = IReal(a,b)
        h = (x.sup-x.inf)/float(n)
        aux = x.inf
        for i in range(n):
                listaInt += [ IReal(aux, aux+h) ]
                aux = aux+h
        for i in range(n):
                totalaux = IReal(0)
                if listaInt[i].inf == 0:
                        totalaux += 0
                else:
                        totalaux += alfa * (beta ** alfa) *
                           listaInt[i].inf ** (alfa - 1) * e **
                           (-((beta * listaInt[i].inf) ** alfa))
                        totalaux += alfa * (beta ** alfa)*
                           listaInt[i].sup ** (alfa - 1) * e **
                           (-((beta * listaInt[i].sup) ** alfa))
                        aux = (listaInt[i].inf+listaInt[i].sup)/2
                        totalaux += 4 * (alfa * (beta ** alfa)*
                           aux ** (alfa - 1) * e ** (-((beta * aux
                           ) ** alfa)))

                totalaux = totalaux * ((listaInt[i].sup-listaInt[
                   i].inf)/6)
                total = totalaux + total
        end_time = time.time()
        print(end_time -begin_time)
        return total
```