Lógica, Apenas Lógica

Renan Aparecido Stuchi*

2019, v-1.0.0

Resumo

Neste artigo pretende-se introduzir uma teoria a respeito da origem de tudo. O objetivo inicial é responder se existe algo ao invés de nada. Essa pergunta vem incomodando a filosofia e a ciência até os dias de hoje. A resposta a essa pergunta está na compreensão de que a lógica em sua essência remete ao nada, que NÃO É, ou seja, nega a si mesmo (nega ser). A negação de si gera expansões binomiais, no qual suas amostras combinadas em cada passo dessa expansão se aproximam da distribuição normal e se aproximam do centro dessa distribuição infinitamente, o que configura o teorema central do limite. Os passos da expansão binomial regidos pela probabilidade descrita no teorema central do limite compreendem a consciência e tornam visíveis o que é e o porquê de seus aspectos mais perceptíveis: infinito, tempo, espaço, gravidade, matéria escura, energia escura e buraco negro. Como essência da expansão binomial, do teorema central do limite e da consciência e seus aspectos tem-se a lógica em sua dualidade, que por um lado NÃO É e por outro É ilógica, imutável e inexistente, uma vez que a existência está em tudo aquilo que NÃO É.

Palavras-chaves: lógica. nada. tudo. binômio. expansão binomial. teorema central do limite. consciência. infinito. tempo. espaço. gravidade. matéria escura. energia escura. buraco negro.

Abstract

This article aims to introduce the theory about the origin of everything. The initial goal is to answer if there is something instead of nothing. This question has been bothering philosophy and science to this day. The answer to this question lies in understanding that logic in its essence refers to nothing, which IS NOT, that is, denies itself (denies being). Self-denial generates binomial expansions, in which their combined samples at each step of this expansion approach the normal distribution and approach the center of this distribution infinitely, which configures the central limit theorem. The steps of binomial expansion governed by the probability described in the central limit theorem comprise consciousness and make visible what it is and why its most noticeable aspects are: infinity, time, space, gravity, dark matter, dark energy, and black hole. The essence of binomial expansion, the central limit theorem and consciousness and its aspects is logic in its duality, which on the one hand IS NOT and on the other hand IS illogical, unchanging and non-existent, since existence is in all that IS NOT.

Keywords: logic. nothing. all. binomial. binomial expansion. central limit theorem. consciousness. infinite. time. space. gravity. dark matter. dark energy. black hole.

*E-mail: ren.stuchi@gmail.com | GitHub: <https://github.com/RenStu/logic>

Introdução

O raciocínio deste texto surgiu como resposta a pergunta mais essencial que a filosofia pode formular e que a ciência até então não foi capaz de responder plenamente, que é se existe algo ao invés de nada ou porque existe algo ao invés de nada? Essa pergunta foi feita pela primeira vez pelo filosofo Gottfried Wilhelm Leibniz em uma carta de 1697 e é frequentemente descrita como a maior questão filosófica (LEIBNIZ, 1697).

A resposta a essa pergunta vem da resposta do que é a lógica. Ao explorar o que a lógica é e o que ela NÃO É, deu origem a uma teoria a respeito da origem de tudo, de todas as coisas. A lógica em sua essência remete ao nada, que NÃO É, ou seja, nega a si (nega ser). A negação de si gera expansões binomiais, no qual suas amostras combinadas em cada passo dessa expansão se aproximam da distribuição normal e se aproximam do centro dessa distribuição infinitamente, o que configura o teorema central do limite. A compreensão destes dois conceitos matemáticos, expansão binomial e teorema central do limite, são essenciais para entendimento dessa teoria.

Os passos da expansão binomial, originados da negação lógica a si, regidos pela probabilidade descrita no teorema central do limite compreendem a consciência e tornam visíveis o que é e o porquê de seus aspectos mais perceptíveis: infinito, tempo, espaço, gravidade, matéria escura, energia escura e buraco negro. Ao responder a pergunta essencial deste estudo também é possível responder as principais questões da ciência, o que é e o porquê é a consciência e seus aspectos, pois são provenientes de um mesmo aspecto lógico.

A lógica NÃO SER é consonante com o NADA, pois se a lógica NÃO É ela também É seu contrário, ou seja, ilógica e imutável. Nessa dualidade, tem-se a existência fundamentada pela lógica que "nega a si", enquanto, por outro lado É ilógica, imutável e inexistente. O texto está disposto na seguinte hierarquia:

1. Lógica
   - 1.1. Expansão binomial
   - 1.2. Teorema central do limite
   - 1.3. Consciência
     - 1.3.1. Infinito
     - 1.3.2. Ondas e subconsciente
     - 1.3.3. Coexistência
     - 1.3.4. Tempo
     - 1.3.5. Espaço
     - 1.3.6. Gravidade
     - 1.3.7. Matéria escura e energia escura
     - 1.3.8. Antimatéria
     - 1.3.9. Buraco negro

Primeiro é definido o que é a lógica e principalmente o que ela NÃO É, assim é apresentado sua consonância ao nada. Depois é descrito como essa lógica primordial, essência de qualquer lógica, se desenvolve gerando novas lógicas por meio de sua expansão binomial. Em seguida é observado que as amostras combinadas em cada passo dessa expansão são regidas pela probabilidade descrita no teorema central do limite o qual da origem ao que a consciência. A negação da nada a si gera as expansões binomiais que são regidas pela probabilidade descrita no teorema central do limite (consciência) é o aspecto lógico responsável em dizer o porquê e o que é o infinito, o tempo, o espaço, a gravidade, a matéria escura, a energia escura e o buraco negro, inicialmente.

1 Lógica

Segundo o dicionário online de Português Dicio(LóGICA..., Porto: 7Graus, 2018), a palavra lógica se refere a:

1. Modo de raciocinar coerente que expressa uma relação de causa e consequência;

2. Maneira coerente através da qual os fatos ou situações se encadeiam.

A palavra lógica expressa uma relação de causa e consequência ou fatos encadeados. Pode-se distinguir como essência dessas duas definições o movimento, a mudança, a transição. A palavra lógica, em sua essência, se encaixa perfeitamente na definição do NADA - NÃO SER. A lógica NÃO SER é consonante com o NADA, pois se a lógica NÃO É ela também É seu contrário, ou seja, ilógica e imutável. Nessa dualidade, tem-se a existência fundamentada pela lógica que "nega a si", enquanto, por outro lado É ilógica, imutável e inexistente. A expressão "negação de si"refere-se à negação do SER - NÃO SER.

A lógica está centrada na mudança e a mudança está centrada naquilo que NÃO É, uma vez que aquilo que É não pode deixar de SER. A mudança demanda que, em algum momento, algo DEIXE DE SER o que fora a se transformar. Em Porfírio (2019b), Parmênides o filósofo da unidade e da identidade do SER, diz que a contínua mudança é a principal característica do não ser. Para Parmênides o SER é uno, eterno, não gerado e imutável.

A lógica SER ilógica não a impede de NÃO SER. Na dualidade SER e NÃO SER, o SER limita e define o NÃO SER *ad infinitum*. É possível se aproximar da definição do SER enumerando e definindo infinitamente tudo o que ele NÃO É.

Figura 1: Analogia da lógica primordial

Reta utilizada para representar e validar o conceito da lógica primordial.

Com base na Figura 1 pode-se extrair as seguintes observações em relação aos pontos **0**, **1** e o **intervalo** entre eles:

Ponto 1 - [1,1] É ilógico, pois é a totalidade não fracionada da reta.

Ponto 0 - [0,0] É ilógico, pois é um ponto nulo incapaz de negar a si, dado que toda lógica ou sublógica (fração lógica) deve se manter negado a si, uma que essa é a premissa básica da lógica. A lógica NÃO É em sua essência, primordialmente.

Intervalo - [0,1[x]0,1] A lógica é possível apenas na representação das frações ou intervalos dos pontos **0** e **1**. Uma fração da reta nega ser a reta, pois é apenas uma parte dela. Os subintervalos são hábeis a negar a si infinitamente, garantindo a premissa primordial da lógica e suas sublógicas, negar a si.

Figura 2: Primeiro momento lógico

Reta fracionada em dois intervalos representando o primeiro momento lógico.

Na Figura 2 a união do traço à reta é a representação de uma divisão lógica, pois é da negação da lógica em SER que surgi esses dois intervalos lógicos ou duas sub-lógicas. O segmento em azul representa a negação da lógica em SER o todo ilógico nesse primeiro momento. As duas frações geradas pela negação lógica negam SER a reta, pois são apenas intervalos delas e são capazes de negar a si infinitamente, garantindo a premissa primordial da lógica NÃO SER.

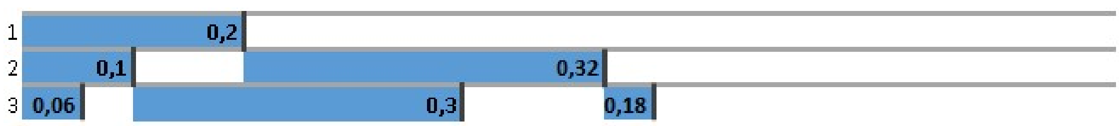
1.1 Expansão binomial

A lógica primordial (negação de si) cria expansões binomiais infinitas. O primeiro momento lógico é o início de uma dessas expansões, porém existem infinitas possibilidades de negação do primeiro momento lógico, o que revela as infinitas expansões binomiais. Uma expansão binomial é análoga a um universo. É importante observar que o primeiro momento lógico negar SER ilógico não transforma o ilógico em lógico, negar não é transformar. É essa dualidade da lógica (SER ilógica e NÃO SER lógica) que garante as infinitas expansões binomiais, uma vez que o ilógico é imutável e por isso pode ser negado infinitamente.

> "Em nossos dias vemos a expansão binomial de uma forma limpa e prática, sendo considerada um dos desenvolvimentos mais lindos e elegantes da matemática, vista com simplicidade em todos os níveis de ensino e pesquisa."(TOGNATO II, 2013).

$$(x+a)^n = \sum_{p=0}^{n} \binom{n}{p} a^p x^{n-p}$$

Figura 3: Momentos lógicos iniciais



Exemplo dos três primeiros momentos de uma expansão.

Com base na Figura 3 pode-se extrair as seguintes observações em relação ao primeiro, segundo e terceiro momentos lógicos:

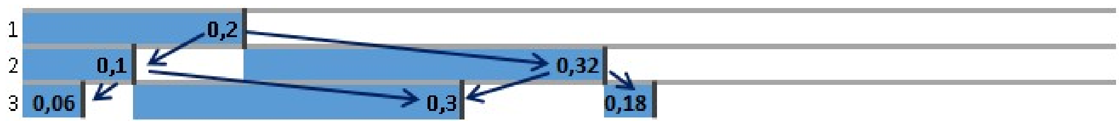
Primeiro momento lógico A negação da lógica primordial a si, a subdivide em duas unidades, que somadas são o todo ilógico. Apesar dessas partes terem proporções diferentes, elas exprimem as mesmas quantidades de pontos ou possibilidades de mudança, uma vez que são representações da lógica primordial, que ad infinitum. A parte fracionada em azul representa a proporção da negação lógica em relação à sua unidade.

Segundo momento lógico É gerado pela negação das duas sub-lógicas primordiais, fracionadas no primeiro momento lógico. Na impossibilidade dessas frações lógicas do primeiro momento lógico continuar negando a si, faria com que elas fossem incapazes de negar suas unidades que formam o todo, ou seja, seriam incapazes de negar as duas unidades e por consequência o todo que é formado precisamente por elas, o que faria da lógica apenas ilógica, uma unicidade. As partes fracionadas em azul representam a proporção da negação lógica em relação à sua respectiva unidade.

Terceiro momento lógico Decorre da negação do segundo momento lógico, assim como o segundo momento lógico decorre da negação do primeiro e assim por diante.

A cada negação ou subnegação da lógica primordial, seus novos valores são influenciados pelos valores adjacentes do momento lógico anterior. Na figura 4, a lógica primordial nega a si gerando o primeiro momento lógico com o valor [0,2]. No segundo momento lógico, suas subdivisões estão contidas no limite imposto pelo valor do primeiro momento lógico. Os pontos do terceiro momento lógico, por exemplo, sofrem as imposições dos valores do segundo momento lógico que por sua vez sofrem a imposição do primeiro. Os valores de momentos lógicos descendentes sofrem imposições acumulativas dos valores dos momentos lógicos anteriores. À imposição de um valor em seus dois valores imediatamente descendentes denominou-se sincronismo lógico. Isso é o que pode ser visto no triângulo de pascal.

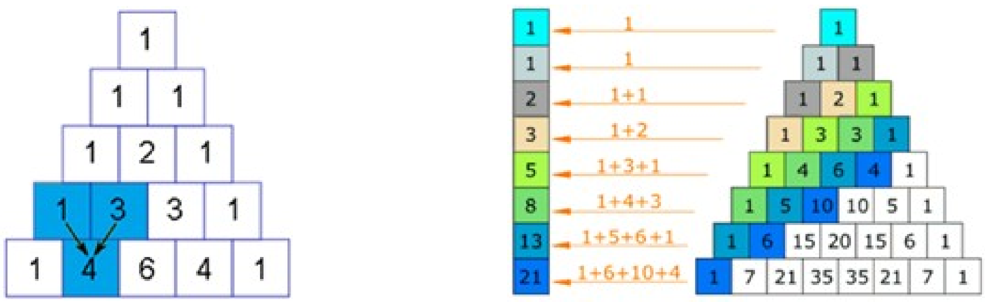
Figura 4: Imposição da expansão binomial



Imposição acumulativa aos momentos lógicos descendentes.

No triângulo de pascal, Figura 5a, cada número é os dois números acima mais próximos somados. Esse número representa quantos diferentes possíveis caminhos levam até ele. Por exemplo, o número [4], na Figura 5a, representa os quatro diferentes caminhos que levam até ele. Um outro aspecto interessante do triângulo de pascal é a sequência de Fibonacci, Figura 5b (PIERCE, 2018b).

Figura 5: Características do triângulo de Pascal



(a) (b)

Fonte: MathsIsFun, 2019.[1]

O NÃO SER da lógica primordial é análogo a uma constante abstrata, ou seja, suas infinitas negações e subnegações transcendem o tempo. Todas essas infinitas negações acontecem no tempo zero. É como dividir algo já subdividido, ou seja, não há intervalo vazio na reta, ao limitar um intervalo os intervalos adjacentes são limitados simultânea e infinitamente. Ao observar simultaneamente as infinitas possibilidades de expansão binomial de um dado intervalo com suas limitações simultâneas e infinitas de cada uma dessas expansões, tem-se um intervalo totalmente limitado, análogo a uma constante (em outras palavras, é todos vezes todos, ou seja, não é preciso esperar o intervalo do momento um, pois todos os intervalos do momento 2 são possíveis para todos os intervalos do momento um e assim por diante). A incapacidade de a lógica negar a si por um intervalo entre suas negações faria a lógica SER ilógica. A lógica é como um algoritmo composto de apenas uma constante auto executada capaz de abstrair a reta, suas divisões e subdivisões, de forma atemporal e intangível. É a consciência que conduz a experiência do tempo, não pela sequência de mudanças que são simultâneas, mas sim pela ordem dessa sequência, que nada mais é que do que a observação das mudanças de cada momento lógico de um determinado intervalo.

Algumas respostas podem deixar essa sequência simultânea (negação de si) mais clara:

Todas as negações acontecem simultaneamente? Sim, infinitas negações na ausência de tempo, ou tempo zero.

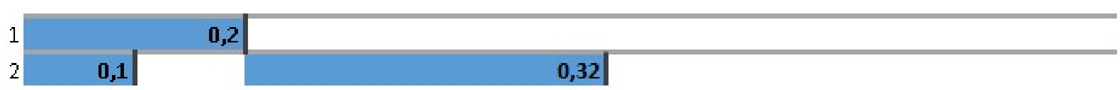
Como ou porque essa simultaneidade acontece? Acontecem em uma sequência de negações de si mesma, no tempo zero, onde em nenhum momento a lógica converte-se em SER, garantindo assim a premissa primordial da constante lógica, NÃO SER.

O que é uma sequência simultânea? É a negação de si (sequência) no tempo zero, ou seja, em nenhum momento a lógica passa a SER durante as infinitas negações (simultaneidade). Sequência simultânea é o sinônimo da constante lógica NÃO SER.

1.2 Teorema central do limite

O primeiro momento lógico divide e gera duas sub-lógicas que negam a si mesmas gerando novas subdivisões ou sub-lógicas que estão presentes no segundo momento lógico. Essa divisão e subdivisões são frações de uma lógica. Assim as sub-lógicas de um momento lógico subnegam o SER, porém unificadas ou somadas elas negam o SER. Assim, a soma dessas frações é a representação de uma unidade lógica. Os momentos lógicos são suas sub-lógicas que representam uma unidade lógica, como são as subunidades de espaço ou tempo (metros e centímetros ou minutos e segundos). Ao dividir uma laranja em duas partes exatas e entregar uma parte para cada pessoa, elas terão uma fração da laranja, mas se entregarmos as duas partes à mesma pessoa ela terá uma laranja em sua totalidade, não duas laranjas, ela terá duas frações da laranja que representa uma laranja.

Figura 6: Momentos lógicos subdivididos

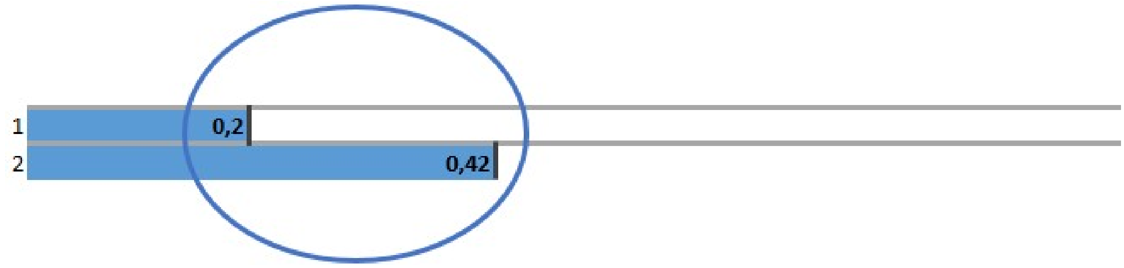


Exemplo dos dois primeiros momentos de uma expansão.

[1] <www.mathsisfun.com/pascal-triangle.html>

E na Figura 7 pode ser observada a representação do primeiro e segundo momentos lógicos como unidades lógicas. A partir de qualquer intervalo, as unidades lógicas de seu segundo momento lógico em diante são obtidas pela soma de suas sub-lógicas. Assim, a unidade do segundo momento lógico do exemplo mostrado na Figura 6, é a soma das subunidades [0,1+ 0,32]. No primeiro momento lógico a lógica em sua unidade nega a si, ou seja, nega SER. Nos demais momentos a união das sub-lógicas ou subdivisões é a representação dessa unidade que nega SER ilógica.

Figura 7: Momentos lógicos unificados

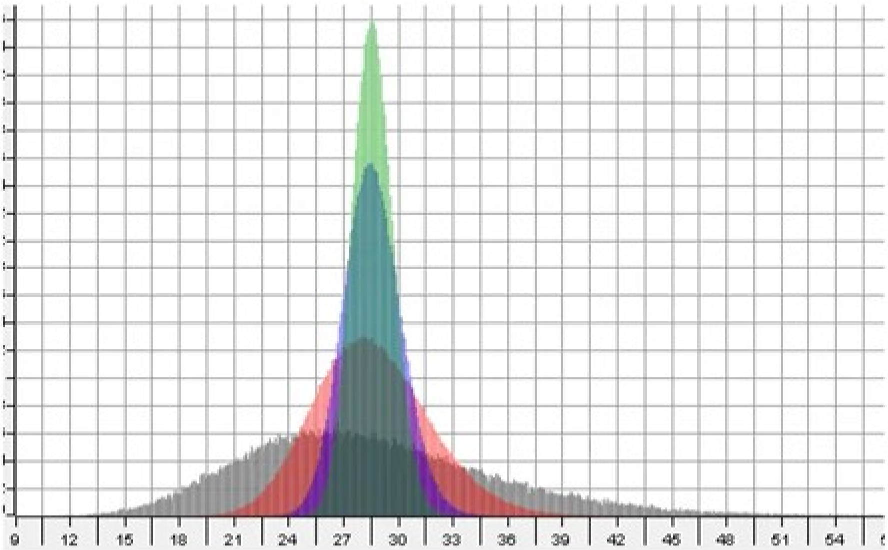


Exemplo dos dois primeiros momentos unificados de uma expansão.

A unificação dos momentos lógicos é a característica que corresponde ao eixo central do teorema central do limite. Esse teorema afirma que a distribuição amostral de uma população se aproxima de uma distribuição normal à medida que as quantidades das amostras aumentam, independente da forma da distribuição da população. Esse fato é especialmente verdadeiro para a quantidade de amostras acima de 30. Um simples teste que demonstra esse fato é o simples lançamento de dados não viciados. Quanto maior for o número de lançamento do dado, maior a probabilidade de o gráfico parecer com o gráfico da distribuição normal (GLEN, 2019). O Apêndice A explica o algoritmo Distribution_PROB com o intuito que clarificar a essência probabilística do teorema central do limite.

A Figura 8 ilustra o fundamento do Teorema central do limite quanto ao fato da aproximação do gráfico ao gráfico da distribuição normal e da aproximação da distribuição da população à mediana à medida que as amostras aumentam. No gráfico são distribuídas 500.000 amostras randomicamente em cada range amostral de ([5-vermelho], [20-azul] e [40-verde]), a cor cinza mostra os valores distorcidos da população (FROST, 2018).

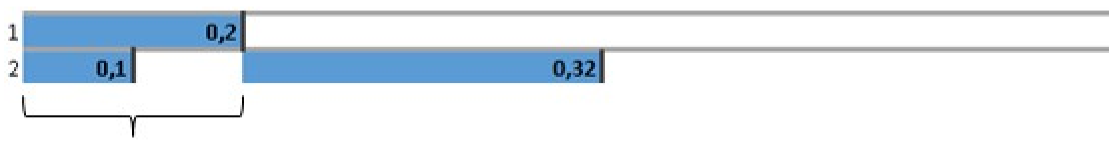
Figura 8: Aproximação do gráfico à distribuição normal e aproximação da distribuição da população à mediana



500.000 amostras distribuídas randomicamente em cada range amostral de ([5-vermelho], [20-azul] e [40-verde] (FROST, 2018)[2].

Quanto maior o número de subunidades lógicas, quanto mais elas ocorrem, maior será o número de amostras dessa população. E acima de 30 amostras, facilmente alcançado pela lógica primordial que tende ao infinito, a tendência da população é se aproximar da distribuição normal e da média populacional infinitamente. A Figura 9 abaixo ilustra a representação de uma amostra no gráfico de expansão binomial usado neste estudo.

Figura 9: Amostra de uma população

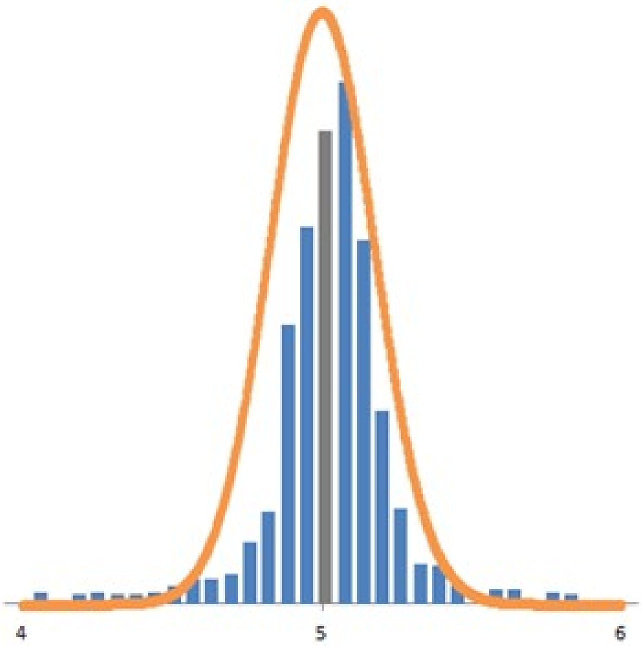


Representação de uma amostra no gráfico de expansão binomial.

É importante notar, conforme Figura 10, que o equilíbrio probabilístico das variações nas faixas a direita e esquerda da mediana, causadas pela distribuição dos momentos lógicos com suas amostras unificadas, podem ilustrar a doutrina dos contrários de Heráclito de Éfeso (PORFíRIO, 2019a).

2 <statisticsbyjim.com/basics/central-limit-theorem>

Figura 10: Equilíbrio probabilístico das amostras contrárias em relação à mediana



1000 momentos lógicos em 500500 amostras distribuídas randomicamente em um range amostral de 1 e 9.

Na Tabela 1 está a probabilidade da distribuição binomial entre 100 a 10000, consonante à amostras unificadas ou médias amostrais tratadas no teorema central do limite. Sua construção se deu com a fórmula da probabilidade binomial geral, que representa uma distribuição uniforme, por meio do algoritmo BinomialDistribuion_PROB clarificado no Apêndice A (PIERCE, 2018a).

$$f(k;n,p) = \binom{n}{k} p^k (1-p)^{n-k}$$

A distribuição binomial se comporta como o lançamento de moedas (cara ou coroa), mas poderia ser utilizado nesse estudo outras distribuições discretas, como o lançamento de dados não viciados, e as observações deste estudo continuariam as mesmas, pois o Teorema central do limite é independente da forma da distribuição da população (FROST, 2018).

Tabela 1: Probabilidade da distribuição binomial

Meta	Soma do Range	Range		Total de Amostras	Amostras do Range	% das Amostras do Range	Range de +/- 14% (28%) da Faixa Central
99,99%	99,994%	31	70	**101**	**39**	**38%**	72,87%
99,99%	99,992%	73	128	**201**	**55**	**27%**	71,11%
99,99%	99,991%	117	184	**301**	**67**	**22%**	72,73%
99,99%	99,990%	162	239	**401**	**77**	**19%**	70,62%
99,99%	99,991%	207	294	**501**	**87**	**17%**	73,64%
99,99%	99,991%	253	348	**601**	**95**	**15%**	72,96%
99,99%	99,991%	299	402	**701**	**103**	**14%**	72,69%
99,99%	99,990%	346	455	**801**	**109**	**13%**	72,69%
99,99%	99,991%	392	509	**901**	**117**	**12%**	72,86%
99,99%	99,991%	439	562	**1001**	**123**	**12%**	73,16%
99,99%	99,991%	486	615	**1101**	**129**	**11%**	73,54%
99,99%	99,991%	533	668	**1201**	**135**	**11%**	71,45%
99,99%	99,991%	580	721	**1301**	**141**	**10%**	72,06%
99,99%	99,990%	628	773	**1401**	**145**	**10%**	72,68%
99,99%	99,991%	675	826	**1501**	**151**	**10%**	73,31%
99,99%	99,990%	723	878	**1601**	**155**	**9%**	71,76%
99,99%	99,991%	770	931	**1701**	**161**	**9%**	72,49%
99,99%	99,990%	818	983	**1801**	**165**	**9%**	73,20%
99,99%	99,990%	866	1035	**1901**	**169**	**8%**	71,90%
99,99%	99,990%	914	1087	**2001**	**173**	**8%**	72,67%
99,99%	99,990%	1394	1607	**3001**	**213**	**7%**	71,86%
99,99%	99,991%	1877	2124	**4001**	**247**	**6%**	72,47%
99,99%	99,990%	2363	2638	**5001**	**275**	**5%**	72,38%
99,99%	99,990%	2850	3151	**6001**	**301**	**5%**	72,75%
99,99%	99,990%	3338	3663	**7001**	**325**	**4%**	72,32%
99,99%	99,990%	3827	4174	**8001**	**347**	**4%**	72,18%
99,99%	99,990%	4316	4685	**9001**	**369**	**4%**	72,23%
99,99%	99,990%	4806	5195	**10001**	**389**	**3%**	72,42%

Tabela gerada pelo algoritmo BinomialDistribuion_PROB com a distribuição binomial de 100 a 10000. [3]

Meta Porcentagem das amostras observadas;

Soma do Range Porcentagem que o **"Range"** atingiu a **"Meta"**, da mediana para as bordas, descentralizado;Porcentagem que o Range atingiu a Meta, da mediana para as bordas, descentralizado;

Range Range de amostras onde a **"Meta"** foi atingida do **"Total de Amostras"**;

Total de Amostras Exibe o range total avaliado, no caso da primeira linha da tabela o valor 101 corresponde às possibilidades de 0 a 100, como se fossem lançadas 100 moedas (distribuição binomial) e somassem suas faces voltadas para cima, podendo ser 0 para as caras e 1 para as coroas. Essa soma é uma combinação de possibilidades não uma permutação, ou seja, na permutação [0 1] é uma possibilidade diferente de [1 0], na combinação essa é uma possiblidade, porém com duas probabilidades de ocorrência;

[3] O Apêndice A é dedicado a clarificar o algoritmo BinomialDistribuion_PROB e validar o fórmula da probabilidade binomial geral usada por ele.

Amostras do Range Quantidade de amostras do **"Range"** do **"Total de Amostras"**;

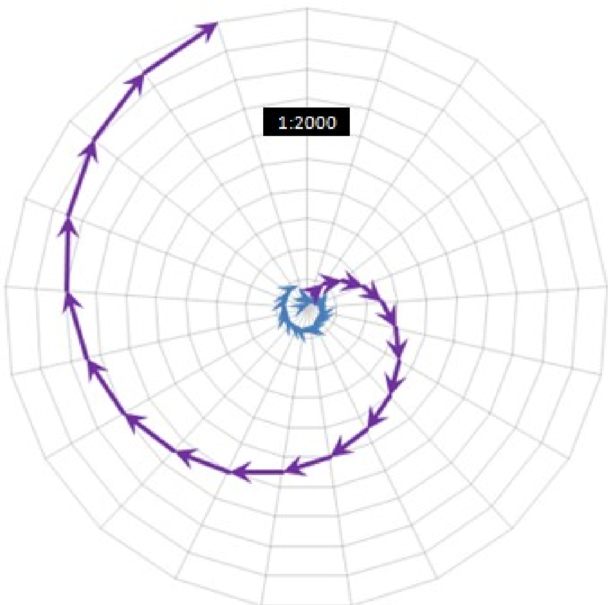
Porcentagem das Amostras do Range Porcentagem que o **"Range** representa do **"Total de Amostras"**;

Range de +/- 14% (28%) da Mediana Esse range é subconjunto do **"Range"**, formado a partir da mediana somando 14% a direita e a esquerda, totalizando 28%. Esses 28% correspondem a aproximadamente 72% das amostras da população do Range, que correspondem a 99,99% da população total. O restante, que representam 72% **"Range"**, correspondem a aproximadamente 28% das amostras. Isso condiz com o Princípio de Pareto também conhecido como a regra do 80/20, que também pode ser 70/30 ou 90/10, por exemplo (TOLEDO, 2014).

Pode-se observar que a medida as amostras aumentam, a porcentagem ocupada por 99,99% das amostras tende a diminuir **"% das Amostras do Range"**, ainda que a quantidade dessas amostras que representam essa porcentagem tenda a aumentar **"Amostras do Range"**.

A coluna de "Amostras do Range", setas azuis no gráfico da Figura 11 vão no sentido ao centro do gráfico, ou seja, apesar de aumentar a quantidade de amostras onde o range das 99,99% das probabilidades se encontram, a proporção que essas amostras assumem no "Total de Amostras"diminuem. As setas em roxo do gráfico representam a distribuição da coluna "Total de Amostras"da Tabela 1. Conforme os momentos lógicos aumentam mais próximos da mediana os 99,99% de suas amostras estarão e mais irrelevantes se tornam os intervalos lógicos mais afastados do centro, os que não fazem mais parte dos 99,99%.

Figura 11: Comparação do total de amostrar com 99,99%



As setas em roxo representam o total das amostras e as em azul os 99,99% [4].

[4] O gráfico da Figura 11 representa as 20 primeiras linhas da Tabela 1, pois sofrem incrementos iguais, de 100 amostras, em cada linha. A linha 21 em diante sofrem incremento de 1000 amostras a cada linha.

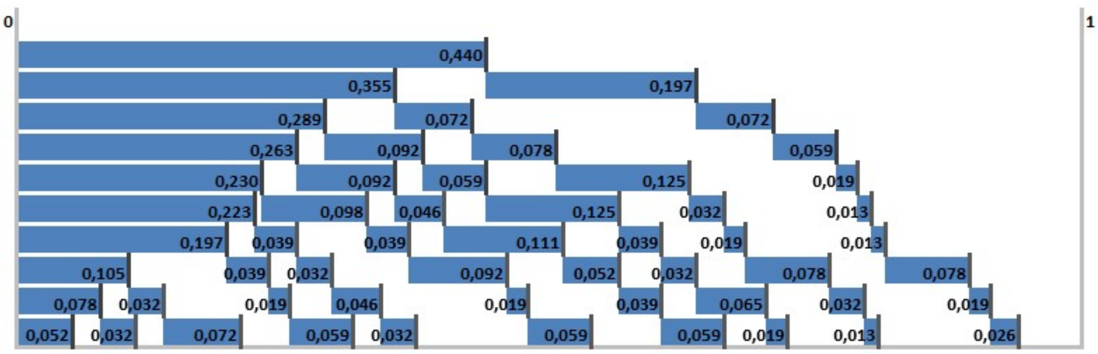
É importante notar na Tabela 1, que quanto mais próximos da mediana os 99,99% das amostras estão, mais amostras precisam adicionadas para que essas amostras do range ocupem uma proporção menor. Por exemplo, as três linhas que representam o “Total de Amostras” de 301, 401 e 501 precisaram de um incremento de 10 amostras “Amostras do Range” em cada linha para que a “ das Amostras do Range” viessem de 22% para 19% e 15% respectivamente, ou seja, a cada 10 amostras incrementas no range a proporção por ele ocupada caia 2%. Porém, ao observar as três linhas que representam o “Total de Amostras” de 7001, 8001 e 9001 que sofreram incremento de 22 amostras “Amostras do Range” em cada linha e ainda sim a “% das Amostras do Range” continua em 4%.

No endereço <https://www.mathsisfun.com/data/quincunx.html> existe uma ferramenta chamada Quincunx ou Galton Board que exemplifica dinamicamente o que as figuras acima mostram. Uma explicação sobre o funcionamento dessa ferramenta pode ser vista em <https://www.mathsisfun.com/data/quincunx-explained.html>.

1.3 Consciência

Como visto na seção do teorema central do limite, um momento lógico é formado por divisão e subdivisões lógicas, como são as subunidades de espaço ou tempo. Um momento lógico (cada linha da Figura 12) é a soma de suas subunidades lógicas, ou seja, uma unidade.

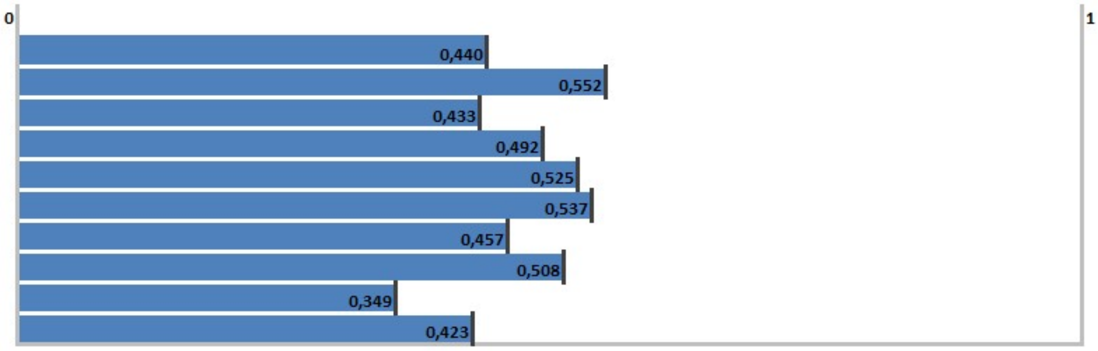
Figura 12: Intervalo lógico



Exemplo de um intervalo lógico com dez momentos lógicos.

A consciência são os momentos lógicos de um intervalo representados em suas unidades.

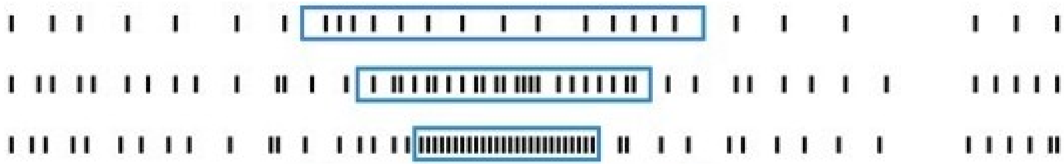
Figura 13: Intervalo lógico consciente



Exemplo de um intervalo lógico consciente com dez unidades de momentos lógicos.

Pode ser observado na Tabela 1 que a probabilidade de 99,99% das amostras, que aumentam em quantidade a medida que crescem os momentos lógicos, tendem a estar cada vez mais ao centro do intervalo lógico, sendo que essa centralização tende ao infinito.

Figura 14: Centralização de 99,99% das amostras



Tendência de centralização do range de 99,99% das amostras.

A consciência é o conjunto dos momentos lógicos de um intervalo. É o aspecto da lógica que unifica as amostras desses momentos, ou seja, é a lógica que abstrai muitos em um, muitas subunidades em uma unidade por momento lógico. Todos os aspectos listados abaixo são inerentes a abstração da lógica chamada consciência.

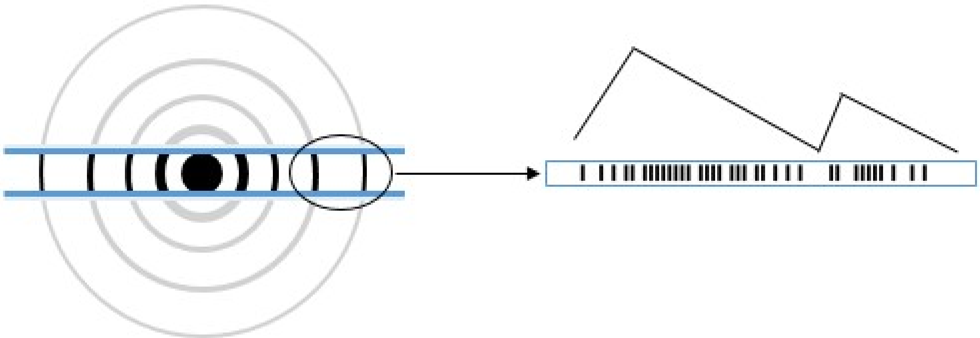
1.3.1 Infinito

Um dos aspectos mais importantes que a negação do nada traz (negação de si), é o infinito, ou seja, em qualquer intervalo lógico cabe o infinito novamente. A lógica primordial que iniciou todo o intervalo lógico é a mesma encontrada em seus intervalos subsequentes. Não há fim, não há meio, apenas infinitos começos. Isso fundamenta como uma lógica de alto nível como a consciência explica a lógica primordial, uma vez que não é preciso voltar ao primeiro momento lógico do intervalo para deduzi-lo, pois esse fenômeno é onipresente em todo o intervalo.

1.3.2 Ondas e subconsciente

Probabilisticamente a distribuição de novas amostras de uma população tendem a concentrar mais amostras sentido a mediana da população com frequências de amostras cada vez maiores neste sentido. Porém, a distribuição dessas amostras com frequências de crescimento uniformes é infinitesimal se comparado às possibilidades randômicas desse crescimento. Assim, a tendência de crescimento dessas frequências sentido a mediana somadas a baixíssima probabilidade (infinitesimal) desse crescimento ser uniforme, conduz a frequências no padrão de ondas.

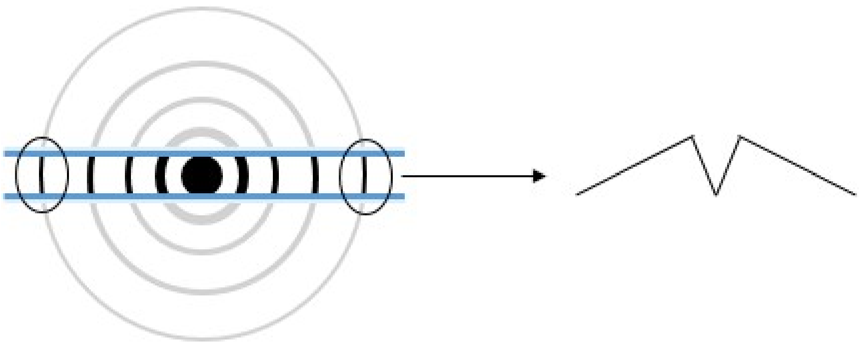
Figura 15: Padrão de onda



Padrão de onda inferido pela tendência dessa distribuição com frequências maiores sentido a mediana da população e a baixíssima probabilidade de crescimento uniforme dessas frequências.

As amostras que mais se parecem em termos de frequências e distribuição são as amostras que fazem parte da mesma onda, que em momentos passados estiveram mais próximas. Elas são frequências opostas não sobrepostas que se completam. Essas ondas formas subconsciência de uma consciência maior. A consciência é única para todo o intervalo, é a lógica do intervalo, enquanto forma subconsciências, como pequenas ondas de uma onda maior. Assim, uma mudança na onda maior (consciência) também é uma mudança na onda menor (subconsciência), mudança essa que é induzida pela subconsciência indiretamente, análogo ao comprimir gás em um cilindro, ao adicionar uma nova molécula de gás no cilindro parcialmente cheio mais próximas ou apertas essas moléculas dentro dele estarão. O contrário também é verdadeiro, uma nova amostra em uma subconsciência, que por esta é observada diretamente é também uma mudança da consciência e vai ser induzida pelas outras subconsciências indiretamente.

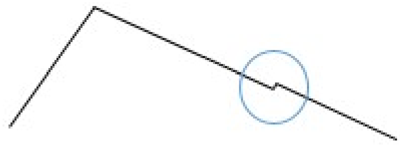
Figura 16: Subconsciência



O padrão de ondas forma subconsciências semelhantes ao padrão criado pela consciência como visto na Figura 8 ou na Figura 10.

Grandes intervalos com baixas frequências de amostras ou grandes intervalos com frequências uniformes de amostras são mais difíceis de observar devido à ausência de grandes discrepâncias. A junção de duas ondas além de eliminar suas discrepâncias, faz com que a primeira onda da união fique maior e a segunda onda acaba por deixar de existir e se tornar parte da primeira que tem seu pico mais próximo da mediana. Probabilisticamente uma onda não morre, apenas une-se com outras ondas mais internas a ela.

Figura 17: Unificação de ondas

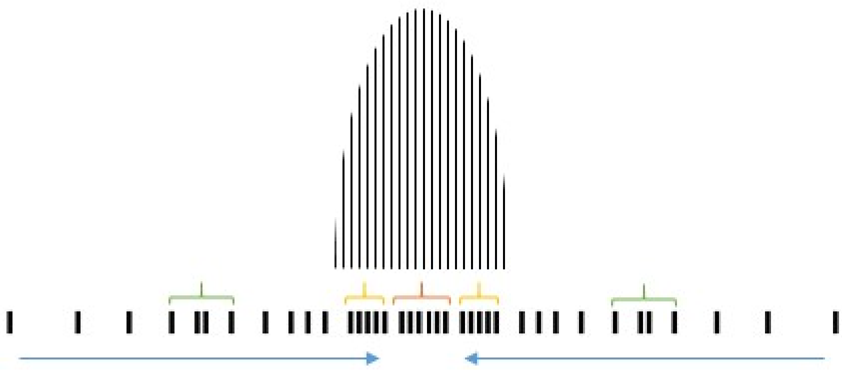


Ondas sendo unificadas para exemplificar o crescimento amostral uniforme.

1.3.3 Coexistência

A tendência probabilística descrita pelo teorema central do limite faz com que a população concentre uma frequência menor de amostras em suas extremidades que vai aumentando gradualmente conforme se aproxima da mediana, conforme visto na seção anterior do padrão de ondas. A frequência de amostras dispostas desta forma torna as interações conscientes mais semelhantes em conjuntos de amostras adjacentes. Assim, ao comparar a frequência de um conjunto de amostras próximo à mediana com um conjunto de amostras mais próximo da extremidade de uma grande população, probabilisticamente haverá uma grande discrepância em suas frequências, ou seja, em um intervalo com tamanho similares haverá mais amostras no conjunto próximo à mediana. Essa observação mostra que conjuntos ou intervalos de frequências adjacentes são mais parecidos do que conjuntos distantes.

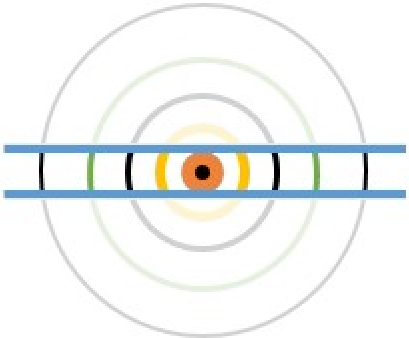
Figura 18: Conjuntos de frequências de um intervalo



Diferentes frequências de um intervalo e suas paridades.

Probabilisticamente as amostras que mais se parecem em termos de frequências e distribuição são as amostras que fazem parte da mesma onda, que em momentos passados estiveram mais próximas. Essas não são frequências opostas que se sobrepõem e se anulam como na antimatéria, mas sim frequências opostas não sobrepostas que se completam.

Figura 19: Intervalos de uma mesma onda

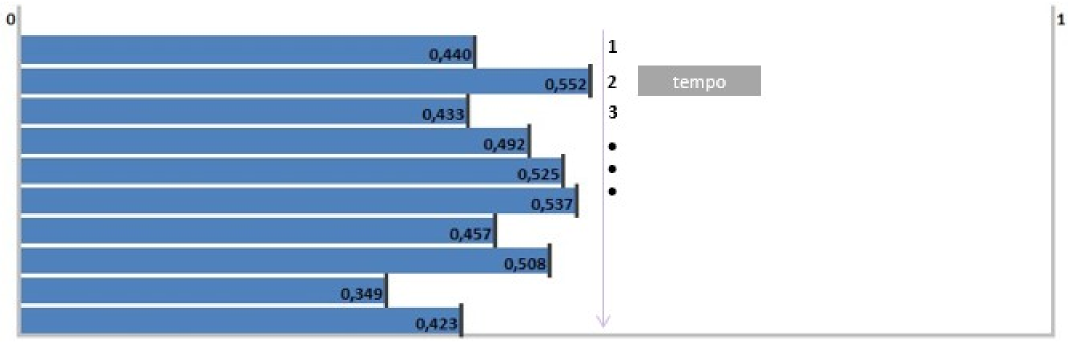


Intervalos opostos não sobrepostos de uma mesma onda que se completam.

1.3.4 Tempo

O tempo é a adição de novos momento lógicos à medida que prossegue a negação desses momentos. Essas mudanças são acumulativas e a medida que aumentam os número de amostras ou momentos lógicos, menos relevante cada nova amostra será dentro do intervalo consciente. Um em cem é mais relevante do que um em mil.

Figura 20: Tempo



Progressão do tempo conforme os momentos lógicos avançam.

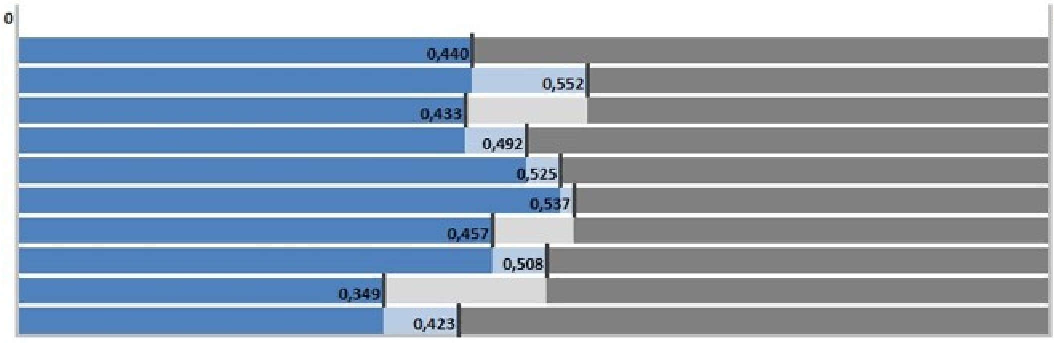
Outro fator importa a observar do tempo é que, probabilisticamente, subconsciências mais próximas da mediana da população terão uma adição maior de novas amostras em seus intervalos, o que são observados diretamente por essas subconsciências. Por outro lado, subconsciências distantes da mediana da população terão uma adição menor de amostras em seus intervalos e sujeitam-se a um número maior de mudança induzidas pela subconsciência indiretamente. Esse fenômeno de observação temporal proporcionado pela consciência e subconsciências evita o paradoxo dos gêmeos (HELERBROCK, 2020).

Na seção Expansão binomial foi apresentado que a lógica é uma sequência de negações de si no tempo zero, ou seja, em nenhum momento entre suas negações a lógica passa a SER, garantindo a premissa primordial da constante lógica NÃO SER. Assim, a lógica é uma sequência infinita e simultânea, uma constante. Logo, o tempo é apenas uma grandeza da consciência oriunda da sequência lógica. A simultaneidade dessa sequência torna a lógica uma constante com todas as suas infinitas possibilidades, sendo esse universo uma delas. Cada universo tem sua sequência de mudanças, que é estática, em uma ordem diferente e é essa ordem que dá origem à grandeza que chamamos de tempo. É essa ordem do universo ou consciência que vai dar a noção do que acontece antes ou depois, ou seja, o passado, o presente e o futuro. Na experiência do tempo conduzida pela consciência a ordenação da sequência é a essência dessa grandeza e, portanto, mais relevante do que sua origem que é de natureza simultânea.

1.3.5 Espaço

O espaço é a relação da proporção dos intervalos dos momentos lógicos. A proporção da fração lógica (intervalo azul) com a unidade lógica (intervalo cinza) e da diferença de entre as frações lógicas (intervalo azul ou cinza claros).

Figura 21: Espaço

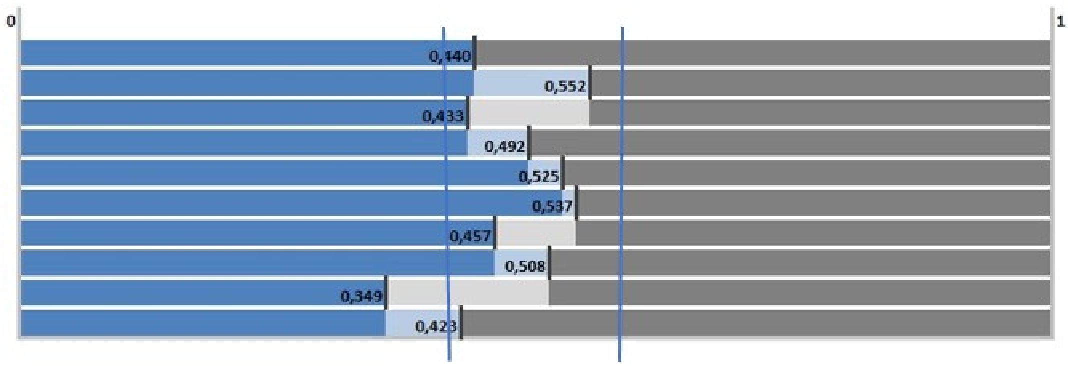


Relação da proporção dos intervalos dos momentos lógicos.

1.3.6 Gravidade

A gravidade é um aspecto probabilístico da distribuição amostral de uma população, como previsto pelo teorema central do limite. Esse teorema afirma que a distribuição amostral de uma população se aproxima de uma distribuição normal à medida que o tamanho das amostras aumentam, o que tende probabilisticamente à centralização infinita das amostras conforme os momentos lógicos progridem. A atração do amor, a gravidade que atraem os objetos à terra e a terra ao sol são sinônimos deste mesmo aspecto. Em outras palavras, a gravidade é análoga a uma conexão que fica cada vez mais forte a medida que se aproxima da mediana, onde tem-se uma concentração de amostras cada vez maior.

Figura 22: Gravidade

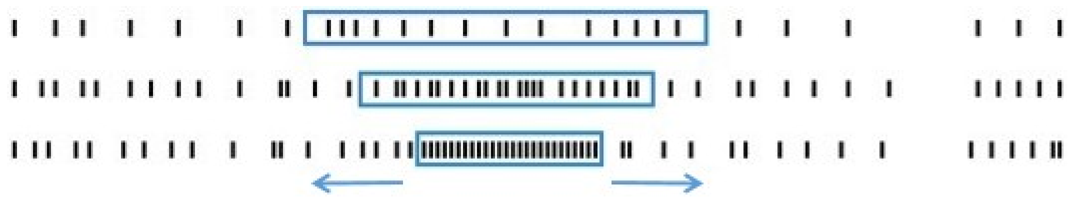


Centralização infinita das amostras conforme os momentos lógicos progridem.

1.3.7 Matéria escura e energia escura

Quanto maior o número de amostras e mais próximas elas estão da mediana, mais elas farão parte dos 99,99% e ainda mais amostras também estarão nos 0,01%, conforme a Tabela 1. Assim, a observação desses 0,01% passa a ser cada vez mais difícil, pois sua relevância consciente passa a ser cada vez mais próxima de zero. É importante notar também que a medida que os 99,99% aumentam em número de amostras, menos relevante cada nova amostra será dentro desse conjunto (um em cem é mais relevante do que um em mil) e uma porcentagem menor será ocupada pelo range dos 99,99% das amostras, conforme a Tabela 1.

Figura 23: Analogia da matéria escura e energia escura

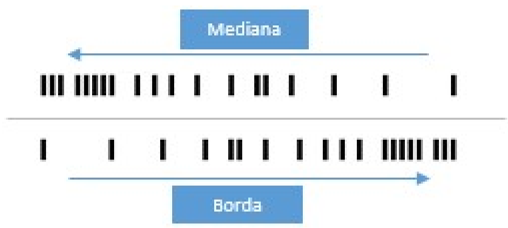


Fenômenos antevistos ou conjecturados pela consciência.

1.3.8 Antimatéria

Independente do intervalo observado (análogo à bytes, kilobytes, prótons, elétrons etc.), que são contextos lógicos de observação e/ou utilização consciente, este pode estar com sua maior concentração de amostras no sentido da mediana, o que é o sentido provável conforme os números de amostras crescem em um intervalo, conforme teorema central do limite. Essas amostras também podem estar com sua concentração no sentido oposto a mediana, porém com uma ocorrência probabilística cada vez menos conforme as amostras aumentam. Na Figura 24 é exibido dois intervalos idênticos com suas amostras com concentrações opostas.

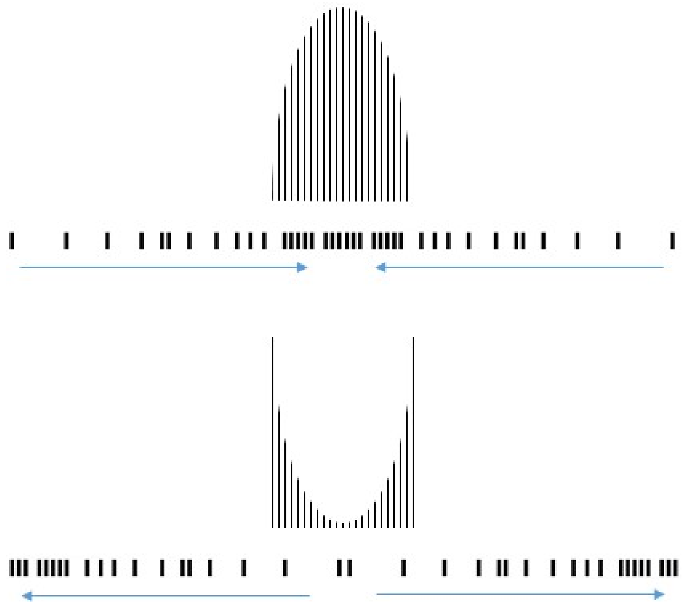
Figura 24: Parte de um intervalo idêntico com suas concentrações de amostras opostas



Parte de um intervalo idêntico distribuídos de formas opostas.

O merge ou soma dos intervalos opostos da Figura 24 os tornaria um intervalo simétrico, ou seja, não estaria em nenhum dos sentidos. Na Figura 25 é exibido um intervalo consciente completo com suas concentrações de amostras sentido à mediana e outro idêntico, mas com suas concentrações sentido às bordas do intervalo.

Figura 25: Intervalos conscientes com suas concentrações de amostras opostas

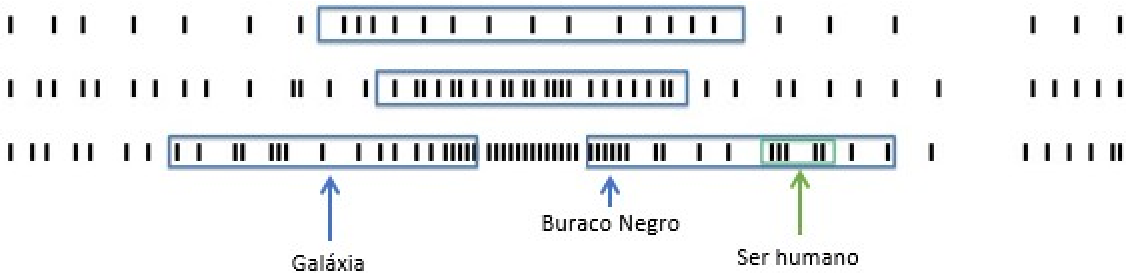


Intervalos conscientes completos e idênticos distribuídos de formas opostas.

1.3.9 Buraco negro

O buraco negro é uma concentração muito alta de amostras, uma altíssima frequência de amostras em um intervalo lógico, que muito provavelmente, em um passado distante dessa consciência esteve muito próximo ao centro dessa consciência, é o pico de uma onda, onde devido ao alto número não há mais discrepância, ou seja, é um pico de onda com um crescimento uniforme e por isso inobservável.

Figura 26: Buraco negro



Centralização infinita das amostras em uma proporção centralizada cada vez menor.

1.4 Observações

Núcleo Este estudo está centrado na expansão binomial e na distribuição aleatória de amostras que combinadas em cada passo dessa expansão se aproximam da distribuição normal e se aproximam do centro dessa distribuição infinitamente.

Rigidez lógica Se a rigidez física e suas leis parecem ser intransponíveis, abaixo dela está à lógica, ainda mais rígida e intransponível, pois fora da lógica o que se tem é o inexistente ilógico. A existência está dentro das possibilidades lógicas.

Matemática A matemática é uma ótima abstração do universo, mas ela não é a linguagem do universo, pois abaixo da matemática está à lógica, a base da matemática e de toda a existência.

Bem e mal O bem e o mal são observações da consciência. Ou seja, se está claro a negação tende a escurecer, se está calor a esfriar etc. É a briga dos contrários de Heráclito de Éfeso. Os contrários tendem a se equilibrarem.

Perfeição A lógica primordial é a mais simples das lógicas, é a essência da existência. Uma lógica tão simples quanto eficiente, tão eficiente quanto é perfeição. A mais poderosa lógica:

Onipotente A essência da existência;

Onisciente Fluxo de todas as abstrações lógicas descendentes;

Onipresente Suas frações estão em toda a existência.

Realidade Como possibilidade lógica o sonho é tão real quando a "realidade". Talvez o estudo das possibilidades lógicas leve a caminhos onde os sonhos possam ser tão reais quanto à realidade, já que os dois não passam de lógica, como sonhos lúcidos (TOLEDO, 2014). Isso talvez explique por que outras possíveis formas de vidas "inteligentes", quando evoluídas, deixam de buscar esse tipo de vida em um possível vasto universo à procurarem dentro de si, onde se pode encontrar algo bem maior que qualquer universo, o infinito.

Foco É provável que o foco da observação consciente esteja nas amostras ao redor e na mediana de uma população, onde está o foco das mudanças. A probabilidade dessa constante centralização, desse "puxar para dentro" tornando mais irrelevantes as amostras à margem talvez faça com que certos animais sintam a experiência do sono, pois esse foco cada vez mais ao centro pode ir desfocando os sentidos mais distantes desse foco.

Considerações Finais

Este é um estudo da lógica que resultou em uma teoria a respeito da origem de tudo. Todas as linhas de raciocínio deste estudo podem ser aprofundadas e detalhadas.

Eventualmente pode ser considerado um estudo filosófico e/ou científico, entretanto a base desses dois importantes ramos é a lógica, o núcleo dessa teoria.

A resposta da pergunta central desse estudo (se existe algo ao invés de nada) vem da lógica em sua dualidade, que por um lado NÃO É e por outro É ilógica, imutável e inexistente, uma vez que a existência está em tudo aquilo que NÃO É. A resposta a essa pergunta está na compreensão de que a lógica em sua essência remete ao nada, que NÃO É, ou seja, nega a si mesmo (nega ser). A negação de si gera expansões binomiais, no qual suas amostras combinadas em cada passo dessa expansão se aproximam da distribuição normal e se aproximam do centro dessa distribuição infinitamente, o que configura o teorema central do limite. Os passos da expansão binomial regidos pela probabilidade descrita no teorema central do limite compreendem a consciência e tornam visíveis o que é e o porquê de seus aspectos mais perceptíveis: infinito, tempo, espaço, gravidade, matéria escura, energia escura e buraco negro.

É importante observar que todos esses aspectos citados são regidos pelo mesmo modelo descrito neste artigo, simplicidade e eficiência em níveis da perfeição.

Que o modelo desse estudo seja o início de uma nova era. Uma era onde o ser humano possa desenvolver a si e observar que é o hospedeiro de inúmeros universos, do infinito. Que essa evolução possa transformar os sonhos em "realidade" e que seja possível observar que a "realidade" não é diferente de um sonho, uma vez que ambas são apenas lógicas.

Pensar que algo físico tenha surgido do nada se faz incoerente com sua natureza nula, ilógica, imutável e inexistente.

Referências

FROST, J. *Central Limit Theorem Explained*. 2018. Website Statistics By Jim. Disponível em: <https://statisticsbyjim.com/basics/central-limit-theorem>. Acesso em: 05 nov 2019. Citado 3 vezes nas páginas 7, 8 e 9.

GLEN, S. *Central Limit Theorem: Definition and Examples in Easy Steps*. 2019. Website Statistics How To. Disponível em: <https://www.statisticshowto.datasciencecentral.com/probability-and-statistics/normal-distributions/central-limit-theorem-definition-examples>. Acesso em: 01 nov 2019. Citado na página 7.

HELERBROCK, R. *Paradoxo dos gêmeos*. 2020. Website Brasil Escola. Disponível em: <https://brasilescola.uol.com.br/fisica/paradoxo-dos-gemeos.htm>. Acesso em: 16 dez 2020. Citado na página 16.

LEIBNIZ, G. W. *SOBRE A ORIGEM FUNDAMENTAL DAS COISAS*. 1697. Leibniz Brasil. Disponível em: <https://leibnizbrasil.pro.br/leibniz-traducoes/sobre-origem-fundamental-das-coisas.htm>. Acesso em: 25 nov 2019. Citado na página 2.

LóGICA. In: DICIO, Dicionário Online de Português. Porto: 7Graus, 2018. Dicionário Online. Disponível em: <https://www.dicio.com.br/logica>. Acesso em: 05 abr 2018. Citado na página 3.

PARKER, D. *BigDecimal - C# implementation of an arbitrary size, arbitrary precision decimal number class, with relevant mathematical operations*. 2018. GitHub - proprietário software. Disponível em: <https://github.com/dparker1/BigDecimal/blob/3e0a4f1ba4c72c0b28d6571fcc6259558be104bd/BigDecimal/BigDecimal.cs>. Acesso em: 27 nov 2019. Citado na página 25.

PIERCE, R. *The Binomial Distribution*. 2018. Website Math is Fun. Disponível em: <http://www.mathsisfun.com/data/binomial-distribution.html>. Acesso em: 05 nov 2019. Citado 2 vezes nas páginas 9 e 22.

PIERCE, R. *Pascal's Triangle*. 2018. Website Math is Fun. Disponível em: <http://www.mathsisfun.com/pascals-triangle.html>. Acesso em: 05 nov 2019. Citado na página 5.

PORFíRIO, F. *Heráclito*. 2019. Website Brasil Escola. Disponível em: <https://brasilescola.uol.com.br/filosofia/heraclito.htm>. Acesso em: 01 nov 2019. Citado na página 8.

PORFíRIO, F. *Parmênides*. 2019. Website Brasil Escola. Disponível em: <https://brasilescola.uol.com.br/filosofia/parmenides.htm>. Acesso em: 01 nov 2019. Citado na página 3.

TOGNATO II, J. O. O binômio de newton. *Departamento de Matemática – UFPR*, p. 10–13, 2013. Disponível em: <http://www.educadores.diaadia.pr.gov.br/arquivos/File/fevereiro2016/matematica_dissertacoes/dissertacao_jose_osvaldo_tognato.pdf>. Acesso em: 05 nov 2019. Citado na página 4.

TOLEDO, M. *Pareto: o mínimo de esforço para o máximo de resultado*. 2014. Website Administradores. Disponível em: <https://administradores.com.br/artigos/pareto-o-minimo-de-esforco-para-o-maximo-de-resultado>. Acesso em: 17 nov 2019. Citado 2 vezes nas páginas 11 e 20.

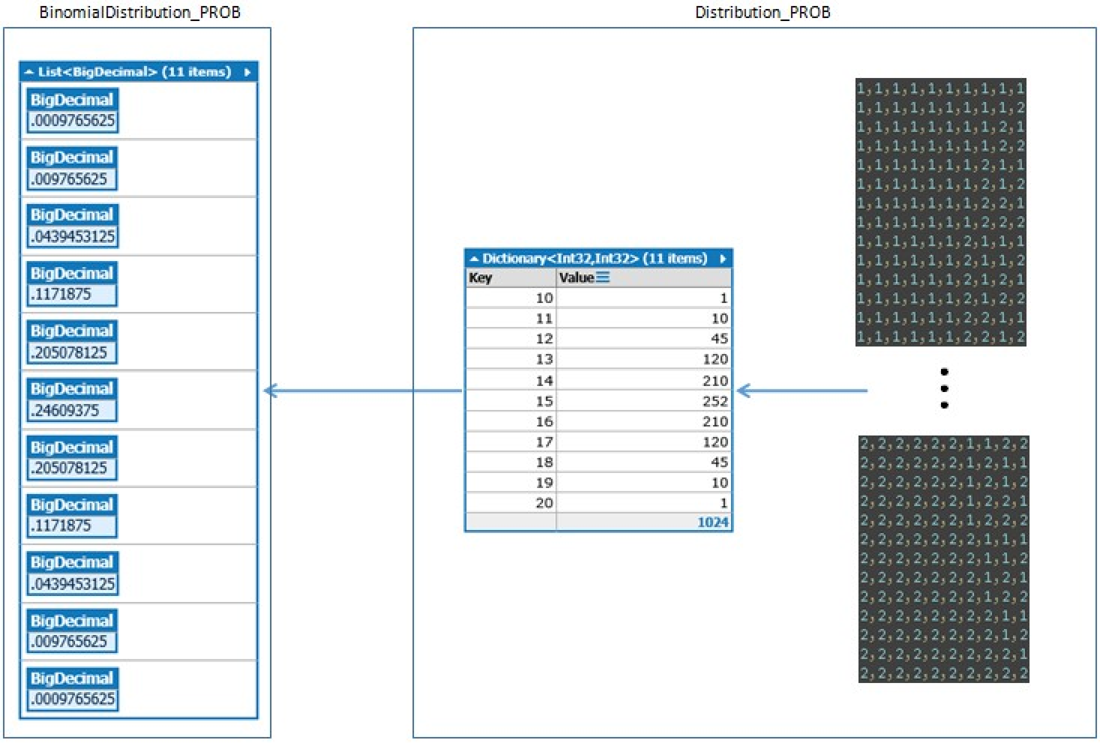
APÊNDICE A – Algoritmos

O algoritmo BinomialDistribution_PROB tem como resultado a probabilidade de distribuição de um range e utiliza a fórmula da probabilidade binomial geral abaixo. Esse algoritmo tem o mesmo resultado do algoritmo Distribution_PROB, porém a execução do BinomialDistribution_PROB é muito mais rápida e tem maior capacidade por usar números grandes como o BigInteger e o BigDecimal. Ambos os algoritmos foram feitos em C# com o LINQPad 5 [5]. Na Figura 27 é mostrado o resultado dos algoritmos para o range de 0 a 10, análogo ao lançamento de 10 moedas ao chão e somado os valores de caras e coras, sendo, por exemplo, a coroa com o valor um e a cara o valor dois. O algoritmo Distribution_PROB soma cada uma das 1024 possibilidades [1,1,1,1,1,1,1,1,1,1 - 1,1,1,1,1,1,1,1,1,2 - ....] e agrupa esses valores somados. No algoritmo Distribution_PROB esse conjunto de possibilidades é um produto cartesiano das possíveis combinações, o que torna esse algoritmo lento rapidamente, porém ele é importante para validar e facilitar o entendimento da a fórmula da probabilidade binomial geral utilizada no algoritmo BinomialDistribution_PROB (PIERCE, 2018a). Na Figura 27, a tabela no interior de Distribution_PROB mostra esse agrupamento e o total de possibilidades, 1024. Ao dividir cada valor agrupado pelo total tem-se o resultado probabilístico alcançado pela fórmula empregada no BinomialDistribution_PROB. Por exemplo, a probabilidade do somatório das 10 moedas lançadas ser 12 é igual a 45/1024, que é 0,0439453125 ou 4,39%.

$$f(k;n,p) = \binom{n}{k} p^k (1-p)^{n-k}$$

[5] O LINQPad 5 é encontrado em <www.linqpad.net> e pode ser utilizado em sua versão livre, Standard edition, sem expiração.

Figura 27: Resultado dos algoritmos BinomialDistribution_PROB e Distribution_PROB



O algoritmo Distribution_PROB tem o intuito que clarificar a essência probabilística do teorema central do limite.

O algoritmo Distribution_PROB também pode ser utilizado para o lançamento de 5 dados de 6 lados ou 6 dados de 5 lados, por exemplo. Como pode ser observado na Figura abaixo, a distribuição das probabilidades no lance dos dados é semelhante à distribuição binomial, das moedas.

Figura 28: Resultados do algoritmo Distribution_PROB

Dictionary<Int32,Int32> (26 items)

Key	Value
5	1
6	5
7	15
8	35
9	70
10	126
11	205
12	305
13	420
14	540
15	651
16	735
17	780
18	780
19	735
20	651
21	540
22	420
23	305
24	205
25	126
26	70
27	35
28	15
29	5
30	1
	7776

(a) 5 dados de 6 lados

Dictionary<Int32,Int32> (25 items)

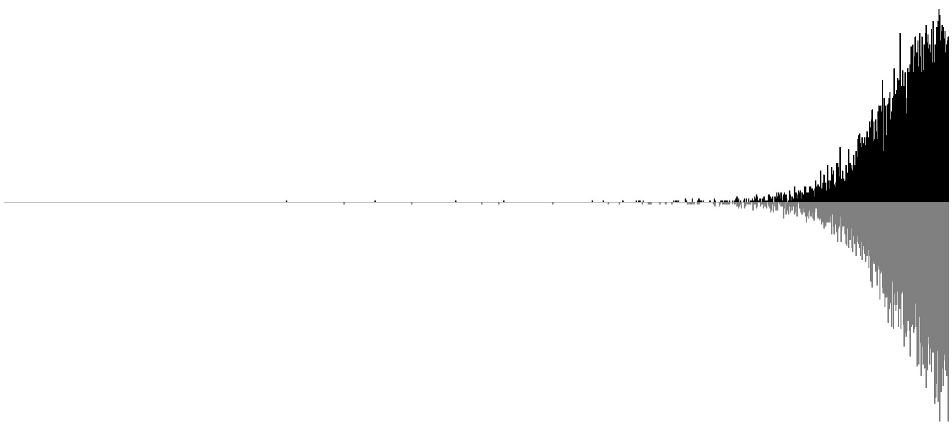
Key	Value
6	1
7	6
8	21
9	56
10	126
11	246
12	426
13	666
14	951
15	1246
16	1506
17	1686
18	1751
19	1686
20	1506
21	1246
22	951
23	666
24	426
25	246
26	126
27	56
28	21
29	6
30	1
	15625

(b) 6 dados de 5 lados

A distribuição das probabilidades no lance dos dados é consonante à distribuição binomial.

O algoritmo Logic_WavePattern tem como resultado a exibição de um histograma que assume o padrão de ondas, quando colocados lado a lado cada uma de suas barras do lado esquerdo e lado direito da mediana. Esse histograma é gerado a partir da randomização de valores conforme Figura 12 e Figura 13, seguindo o Teorema central do limite.

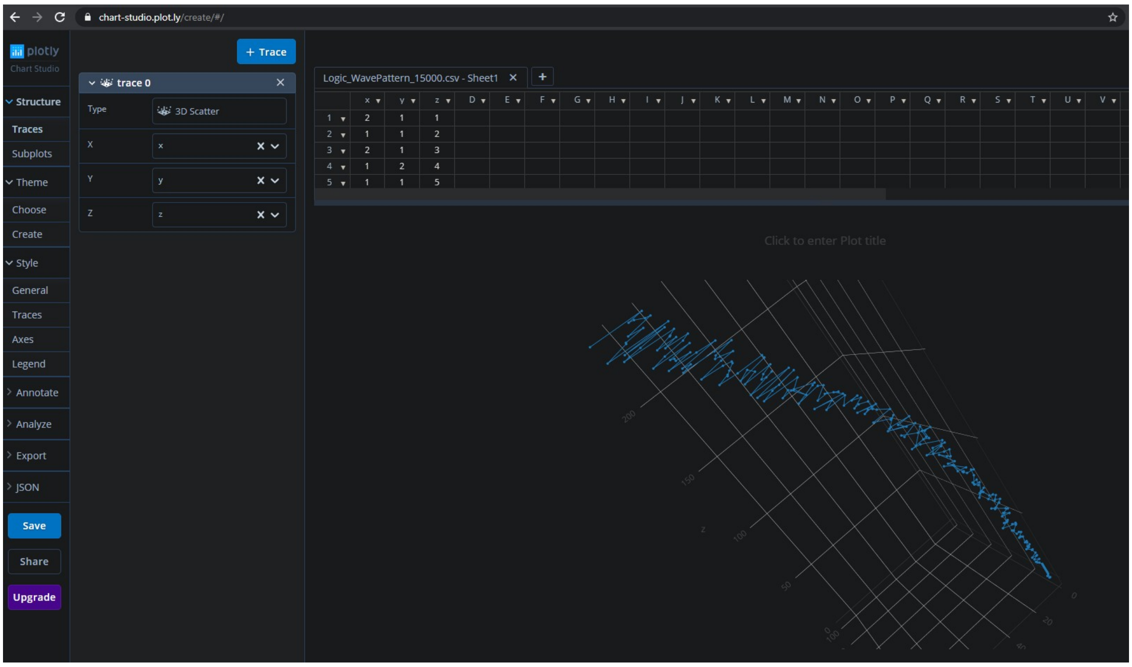
Figura 29: Histograma em padrão de ondas do algoritmo Logic_WavePattern



Resultado gerado randomicamente e exibido pelo algoritmo Logic_WavePattern.

Outro resultado do algoritmo Logic_WavePattern é obtido a partir do console do LINQPad 5, onde se tem como saída um arquivo do tipo .csv que pode ser importado no Chart Studio da plotly <https://chart-studio.plot.ly/create> para geração de um gráfico de dispersão 3D. O mais importante do gráfico são os pontos que representam a matéria gerada a partir das ondas pelas três coordenadas do espaço, as linhas são usadas para facilitar a visualização das aspirais que já começam a se formar mesmo como volumes muito baixo de dados.

Figura 30: Gráfico de dispersão 3D do algoritmo Logic_WavePattern



O exemplo pode ser acessado em: <https://chart-studio.plot.ly/create/?fid=ren.stuchi:5&fid=ren.stuchi:4>.

BinomialDistribution_PROB [Code]

Para execução deste trecho de código é necessário a implementação do BigDecimal, um exemplo dessa implementação, pode ser observado, obedecendo os direitos de licença de software proprietários em (PARKER, 2018). Este estudo não distribui e nem se responsabiliza pela porção do código referente à implementação do BigDecimal, ficando essas responsabilidades à cargo do executor deste trecho de software.

```
//https://www.mathsisfun.com/data/quincunx-explained.html
void Main()
{
    BinomailDistribuition.Possibilities = 10;
    var results = new List<BigDecimal>();
    results.Load();
    results.Print(true); //send false to print Table 1.
}

public static class BinomailDistribuition
{
    public static int Possibilities = 0;
    static int middleLeft = 0;
    static int middleRight = 0;
    static int resultCount = 0;

    public static void Load(this List<BigDecimal> results)
```

```
    {
        for (int i = 0; i <= Possibilities; i++)
        {
            var fatorLeft = Fatorial(Possibilities);
            var fatorRight = BigInteger.Multiply(Fatorial(i), Fatorial(Possibilities - i));
            BigInteger fat = BigInteger.Divide(fatorLeft, fatorRight);
            var powLeft = new BigDecimal(1, 0, 1000000000);
            var powRight = new BigDecimal(1, 0, 1000000000);
            if (i != 0)
                powLeft = BigDecimal.Pow(new BigDecimal(5, 1, 1000000000), i);
            if (i != Possibilities)
                powRight = BigDecimal.Pow(new BigDecimal(5, 1, 1000000000), (Possibilities -
                    i));
            var prob = new BigDecimal(fat) * powLeft * powRight;
            results.Add(prob);
        }
    }

    public static BigInteger Fatorial(int value)
    {
        BigInteger fatorial = 1;
        for (int n = 1; n <= value; n++)
        {
            fatorial *= n;
        }
        return fatorial;
    }

    public static void Print(this List<BigDecimal> results, bool printTableProbability)
    {
        if (!printTableProbability)
        {
            var sum = results.Sum();
            var middle = (middleRight - middleLeft) / 2;
            var middlePercent = ((middleRight - middleLeft) * 14) / 100;
            var list = results.Where((x, i) => i >= middleLeft && i <= middleRight).ToList();
            var listPareto = list.Where((x, i) => i >= (middle - middlePercent) && i <=
                (middle + middlePercent)).ToList();
            var percentOfSum = (middleRight - middleLeft) * 100 / resultCount;
            var sumPercent = sum * new BigDecimal(100, 0, 1000000000);
            var paretoResult = new BigDecimal(0, 0, 1000000000);
            listPareto.ForEach(x => { paretoResult = paretoResult + x; });

            sumPercent.Dump("sum");
            middleLeft.Dump("middleLeft");
            middleRight.Dump("middleRight");
            (middleRight - middleLeft).Dump("itens of sum");
            percentOfSum.Dump("percent of sum");
            resultCount.Dump("total");
            paretoResult.Dump("20/80");
        }
        else
        {
            results.Dump(); //Valid Binomial distribution
        }
    }

    public static BigDecimal Sum(this List<BigDecimal> results)
    {
        resultCount = results.Count;
        middleLeft = resultCount / 2;
        middleRight = middleLeft * 2 < resultCount ? middleLeft + 1 : middleLeft;

        var sum = middleLeft != middleRight ? results[middleLeft] + results[middleRight] :
            results[middleRight];
        while ((sum * new BigDecimal(100, 0, 1000000000)) < new BigDecimal(9999, 2,
            1000000000))
        {
            middleLeft--;
            middleRight++;
            if (middleLeft >= 0)
                sum = sum + results[middleLeft];
            if (middleRight <= Possibilities)
                sum = sum + results[middleRight];
        }
        return sum;
    }
}

//Exemple of BigDecimal class - https://github.com/dparker1/BigDecimal/blob/
//3e0a4f1ba4c72c0b28d6571fcc6259558be104bd/BigDecimal/BigDecimal.cs
```

Distribution_PROB [Code]

```
//https://exercicios.brasilescola.uol.com.br/exercicios-matematica/
//exercicios-sobre-probabilidade-condicional.htm#questao-1
void Main()
{
    var dice = 2; //Binomial distribution, dice = 2;
    var events = 10;
    var sampling = Math.Pow(dice, events);
    var cartesianProduct = dice.ToArrays(events).CartesianProduct();
    cartesianProduct.PrintGroup(events, dice);
}

public static class CartesianProductContainer
{
    public static IEnumerable<IEnumerable<int>> CartesianProduct(this
         IEnumerable<IEnumerable<int>> sequences)
    {
        IEnumerable<IEnumerable<int>> emptyProduct = new[] { Enumerable.Empty<int>() };
        var result = sequences.Aggregate(
            emptyProduct,
            (accumulator, sequence) =>
                from accseq in accumulator
                from item in sequence
                select new[] { accseq.Concat(new[] { item }).Sum() });

        return result;
    }

    public static IEnumerable<List<int>> ToArrays(this int dice, int events)
    {
        var result = new List<List<int>>();
        for (int j = 1; j <= events; j++)
        {
            var array = new List<int>();
            for (int i = 1; i <= dice; i++)
                array.Add(i);

            result.Add(array);
        }

        return result;
    }

    public static void PrintGroup(this IEnumerable<IEnumerable<int>> list, int events, int
         dice)
    {
        var listCountDict = Enumerable.Range(1, dice * events).ToDictionary(x => x);
        Group(listCountDict, list);
        listCountDict.Dump("Values");
    }

    public static void Group(Dictionary<int, int> dict, IEnumerable<IEnumerable<int>> list)
    {
        foreach (var key in dict.Keys.ToList())
            dict[key] = 0;

        foreach (var item in list)
            dict[item.First()]++;

        var zeroKey = 0;
        foreach (var item in dict)
            if (item.Value == 0)
                zeroKey = item.Key;
            else continue;

        for (int i = 1; i <= zeroKey; i++)
            dict.Remove(i);
    }
}
```

Logic_WavePattern [Code]

```
//http://csharphelper.com/blog/2015/09/draw-a-simple-histogram-in-c/
//https://github.com/naudio/NAudio.WaveFormRenderer
[STAThread]
void Main()
{
  Application.EnableVisualStyles();
  Application.Run(new MainForm());
}

public partial class MainForm : Form
{
  public MainForm()
  {
    InitializeComponent();
  }
  //###########################################################################
  private const int LENGHT = 15000;
  private const int GROUP = 10;
  private bool nestedHistogram = false;
  //###########################################################################
  private double m_dZoomscale = 1.0;
  public static double s_dScrollValue = .05;
  private Point MouseDownLocation;
  private Matrix transform = null;
  private NumbsOfCentralLimitTheorem.HistogramResult histogramResult = null;
  private bool printed = false;

  // Make random histogram data.
  private void MainForm_Load(object sender, EventArgs e)
  {
    histogramResult = GetHistogramOfCentralLimitTheorem(LENGHT, GROUP);

    // Make a transformation to the PictureBox.
    RectangleF data_bounds = new RectangleF(0, 0, histogramResult.Size,
         histogramResult.MaxValue * 2);
    PointF[] points =
    {
        new PointF(0, pictHistogram.ClientSize.Height),
        new PointF(pictHistogram.ClientSize.Width, pictHistogram.ClientSize.Height),
        new PointF(0, 0)
      };
    transform = new Matrix(data_bounds, points);
  }

  private void pictHistogram_Paint(object sender, PaintEventArgs e)
  {
    DrawHistogram(e.Graphics, pictHistogram.BackColor, histogramResult,
      pictHistogram.ClientSize.Width, pictHistogram.ClientSize.Height);
  }

  private void pictHistogram_Resize(object sender, EventArgs e)
  {
    pictHistogram.Refresh();
  }

  // Draw a histogram.
  private void DrawHistogram(Graphics gr, Color back_color,
    NumbsOfCentralLimitTheorem.HistogramResult histogramResult, int width, int height)
  {
    PrintResult();
    gr.Clear(back_color);
    gr.Transform = transform;
    gr.ScaleTransform((float)m_dZoomscale, (float)m_dZoomscale);
    FillRectangle(gr, Color.Black, histogramResult.Up, histogramResult.MaxValue, false);
    FillRectangle(gr, Color.Gray, histogramResult.Down, histogramResult.MaxValue, true);
  }

  private void PrintResult()
  {
    if (!printed)
    {
      printed = true;
      var listTuple = new List<(float x, float y, float z)>();
      float previousValueOfZ = 0;
      for (int i = 0; i < histogramResult.Up.Count(); i++)
      {
        if (histogramResult.Up[i] != 0.0001f && histogramResult.Down[i] != 0.0001f)
        {
          if (histogramResult.Up[i] % 1 == 0)
            previousValueOfZ = (int)(previousValueOfZ + 1f);
          else
```

```
          previousValueOfZ += 0.1f;
        var tuple = (x: histogramResult.Up[i], y: histogramResult.Down[i], z:
            previousValueOfZ);
        listTuple.Add(tuple);
      }
    }
    Console.WriteLine("x,y,z");
    foreach (var tuple in listTuple)
      Console.WriteLine(tuple.x.ToString() + "," + tuple.y.ToString() + "," +
          tuple.z.ToString());
  }
}

protected void FillRectangle(Graphics gr, Color color, float[] arrayValues, float maxValue,
    bool down)
{
  using (Pen thin_pen = new Pen(color, 0))
  {
    for (int i = 0; i < histogramResult.Down.Length; i++)
    {
      RectangleF rect;
      if (!down)
        rect = new RectangleF(i, maxValue, 1, arrayValues[i]);
      else
        rect = new RectangleF(i, maxValue - arrayValues[i], 1, arrayValues[i]);
      using (Brush the_brush = new SolidBrush(color))
      {
        gr.FillRectangle(the_brush, rect);
        gr.DrawRectangle(thin_pen, rect.X, rect.Y, rect.Width, rect.Height);
      }
    }
  }
}

protected void pictHistogram_OnMouseWheel(object sender, MouseEventArgs mea)
{
  pictHistogram.Focus();
  if (pictHistogram.Focused == true && mea.Delta != 0)
    ZoomScroll(mea.Location, mea.Delta > 0);
}

private void ZoomScroll(Point location, bool zoomIn)
{
  transform.Translate(-location.X, -location.Y);
  if (zoomIn)
    m_dZoomscale = m_dZoomscale + s_dScrollValue;
  else
    m_dZoomscale = m_dZoomscale - s_dScrollValue;
  transform.Translate(location.X, location.Y);
  pictHistogram.Invalidate();
}

private void pictHistogram_MouseDown(object sender, MouseEventArgs e)
{
  if (e.Button == System.Windows.Forms.MouseButtons.Left)
    MouseDownLocation = e.Location;
}

private void pictHistogram_MouseMove(object sender, MouseEventArgs e)
{
  if (e.Button == System.Windows.Forms.MouseButtons.Left)
  {
    transform.Translate((e.Location.X - MouseDownLocation.X)
      / 40, (e.Location.Y - MouseDownLocation.Y) / 40, MatrixOrder.Append);
    this.Refresh();
  }
}

private NumbsOfCentralLimitTheorem.HistogramResult GetHistogramOfCentralLimitTheorem(int
    length, int group)
{
  var numbsOfCentralLimitTheorem = new NumbsOfCentralLimitTheorem();
  numbsOfCentralLimitTheorem.NestedHistogram = nestedHistogram;
  numbsOfCentralLimitTheorem.RandomResult(length);
  return numbsOfCentralLimitTheorem.GenerateHistogram(group);
}
}

partial class MainForm
{
  private System.ComponentModel.IContainer components = null;

  protected override void Dispose(bool disposing)
```

```
  {
    if (disposing && (components != null))
      components.Dispose();
    base.Dispose(disposing);
  }

  private void InitializeComponent()
  {
    this.pictHistogram = new System.Windows.Forms.PictureBox();
    ((System.ComponentModel.ISupportInitialize)(this.pictHistogram)).BeginInit();
    this.SuspendLayout();

    // pictHistogram
    this.pictHistogram.Anchor =
         ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top
           | System.Windows.Forms.AnchorStyles.Bottom)
           | System.Windows.Forms.AnchorStyles.Left)
           | System.Windows.Forms.AnchorStyles.Right)));
    this.pictHistogram.BackColor = System.Drawing.Color.White;
    this.pictHistogram.Cursor = System.Windows.Forms.Cursors.Cross;
    this.pictHistogram.Location = new System.Drawing.Point(8, 6);
    this.pictHistogram.Name = "pictHistogram";
    this.pictHistogram.Size = new System.Drawing.Size(550, 250);
    this.pictHistogram.TabIndex = 1;
    this.pictHistogram.TabStop = false;
    this.pictHistogram.Resize += new System.EventHandler(this.pictHistogram_Resize);
    this.pictHistogram.Paint += new
         System.Windows.Forms.PaintEventHandler(this.pictHistogram_Paint);
    this.pictHistogram.MouseWheel += new
         System.Windows.Forms.MouseEventHandler(this.pictHistogram_OnMouseWheel);
    this.pictHistogram.MouseDown += new
         System.Windows.Forms.MouseEventHandler(this.pictHistogram_MouseDown);
    this.pictHistogram.MouseMove += new
         System.Windows.Forms.MouseEventHandler(this.pictHistogram_MouseMove);

    // MainForm
    this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
    this.ClientSize = new System.Drawing.Size(563, 262);
    this.Controls.Add(this.pictHistogram);
    this.Name = "MainForm";
    this.Text = "Logic_WavePattern";
    this.Load += new System.EventHandler(this.MainForm_Load);
    ((System.ComponentModel.ISupportInitialize)(this.pictHistogram)).EndInit();
    this.ResumeLayout(false);
  }

  internal System.Windows.Forms.PictureBox pictHistogram;
}

public class NumbsOfCentralLimitTheorem
{
  public float[] ResultList { get; set; }
  public int ResultLength { get; set; }
  public float[] LastList { get; set; }
  public float[] CurrentList { get; set; }
  public int SizeLastList { get; set; }
  public Dictionary<int, float> Histogram { get; set; }
  public bool NestedHistogram { get; set; }
  private int nestedCountDown = 2;

  public NumbsOfCentralLimitTheorem()
  {
    NestedHistogram = false;
    SizeLastList = 2;
    StartLastList();
    StartCurrentList();
  }

  public float[] RandomResult(int length)
  {
    ResultLength = length;
    ResultList = new float[length];
    Random rnd = new Random();
    for (int x = 0; x < length; x++)
    {
      float lineSum = 0;
      for (int i = 1; i < SizeLastList; i++)
      {
        var lastValueLeft = LastList[i - 1];
        var lastValueRight = LastList[i];
        var rndValue = (float)rnd.NextDouble(lastValueLeft, lastValueRight);
        lineSum = lineSum + (rndValue - lastValueLeft);
```

```
      CurrentList[i] = rndValue;
    }
    if (lineSum != 0)
      ResultList[x] = lineSum;
    SizeLastList++;
    LastList = CurrentList;
    StartCurrentList();
  }
  return ResultList;
}

public HistogramResult GenerateHistogram(int group)
{
  Histogram = new Dictionary<int, float>();
  var minValue = ResultList.Min();
  var maxValue = ResultList.Max();
  var rangeValue = maxValue - minValue;
  var amountOfGroups = ResultLength / group;
  var intervalValue = rangeValue / amountOfGroups;
  foreach (var value in ResultList)
  {
    int key = (int)(value / intervalValue);
    if (!Histogram.ContainsKey(key))
      Histogram[key] = 0;
    Histogram[key]++;
  }
  var histogramResult = HistogramResult.Get(Histogram);
  if (NestedHistogram)
    printMaxInterval(histogramResult, Histogram, intervalValue, group);
  return histogramResult;
}

public Dictionary<int, float> GenerateHistogram(List<float> ResultList, KeyValuePair<int,
    float> keyValue, int group)
{
  Histogram = new Dictionary<int, float>();
  var minValue = ResultList.Min();
  var maxValue = ResultList.Max();
  var rangeValue = maxValue - minValue;
  var amountOfGroups = ResultList.Count / group;
  var intervalValue = rangeValue / amountOfGroups;
  foreach (var value in ResultList)
  {
    int key = (int)(value / intervalValue);
    if (!Histogram.ContainsKey(key))
      Histogram[key] = keyValue.Value;
    Histogram[key]-= (1.0F / group);
  }
  return Histogram;
}

private void printMaxInterval(HistogramResult histogramResult, Dictionary<int, float>
    histogram, float intervalValue, int group)
{
  var histogramOrdered = histogram.OrderBy(x => x.Key);
  var middle = histogram.Count / 2;
  var valueUp = histogramOrdered.ElementAt(middle - nestedCountDown);
  var valueDown = histogramOrdered.ElementAt(middle + nestedCountDown);
  var listUp = GenerateList(valueUp, intervalValue);
  var listDown = GenerateList(valueDown, intervalValue);
  var histogramUp = GenerateHistogram(listUp, valueUp, group);
  var histogramDown = GenerateHistogram(listDown, valueDown, group);
  listUp = histogramUp.Values.ToList();
  listDown = histogramDown.Values.ToList();
  var listCountMin = listUp.Count > listDown.Count ? listDown.Count : listUp.Count;
  histogramResult.Up = RefreshArray(histogramResult.Up, listUp, listCountMin);
  histogramResult.Down = RefreshArray(histogramResult.Down, listDown, listCountMin);
}

private float[] RefreshArray(float[] array, List<float> newItens, int listCountMin)
{
  var newArray = new float[array.Count() + listCountMin];
  var rangeValueListMin = newArray.Count() - nestedCountDown - listCountMin;
  var rangeValueListMax = newArray.Count() - nestedCountDown;
  for (int i = 1; i <= nestedCountDown; i++)
    newArray[newArray.Count() - i] = array[array.Count() - i];
  for (int i = 0; i < newArray.Count() - nestedCountDown; i++)
  {
    if (i >= rangeValueListMin && i <= rangeValueListMax)
      newArray[i] = newItens[i - rangeValueListMin];
    else
      newArray[i] = array[i];
  }
```

```
    return (float[])newArray.Clone();
  }

  private List<float> GenerateList(KeyValuePair<int, float> keyValue, float intervalValue)
  {
    var minValueInterval = keyValue.Key * intervalValue;
    var maxValueInterval = minValueInterval + intervalValue;
    var internalList = new List<float>();
    foreach (var value in ResultList)
    {
      if (value >= minValueInterval && value <= maxValueInterval)
        internalList.Add(value);
    }
    return internalList;
  }

  private void StartCurrentList()
  {
    var sizeCurrentList = SizeLastList + 1;
    CurrentList = new float[sizeCurrentList];
    CurrentList[0] = 0;
    CurrentList[sizeCurrentList - 1] = float.MaxValue / 2;
  }

  private void StartLastList()
  {
    LastList = new float[SizeLastList];
    LastList[0] = 0;
    LastList[SizeLastList - 1] = float.MaxValue / 2;
  }

  public class HistogramResult
  {
    public int Size { get; set; }
    public float MaxValue { get; set; }
    public float[] Up { get; set; }
    public float[] Down { get; set; }

    public static HistogramResult Get(Dictionary<int, float> histogram)
    {
      var histogramOrdered = histogram.OrderBy(k => k.Key);
      var result = new HistogramResult();
      var lengthOdd = histogram.Count % 2 > 0;
      var middle = histogram.Count / 2;
      var middleValue = histogramOrdered.ElementAt(middle).Key;
      result.Size = middleValue;
      result.MaxValue = histogram.OrderBy(k => k.Value).Last().Value;
      result.Up = ArrangeArray(new float[middleValue]);
      result.Down = ArrangeArray(new float[middleValue]);
      for (int i = 0; i < middle; i++)
      {
        var keyValue = histogramOrdered.ElementAt(i);
        result.Up[keyValue.Key] = keyValue.Value;
      }
      for (int i = lengthOdd ? middle + 2 : middle + 1; i < histogram.Count; i++)
      {
        var totalValue = middleValue * 2;
        var keyValue = histogramOrdered.ElementAt(i);
        result.Down[totalValue - keyValue.Key] = keyValue.Value;
      }
      return result;
    }

    private static float[] ArrangeArray(float[] array)
    {
      for (int i = 0; i < array.Length; i++)
        array[i] = 0.0001F;
      return array;
    }
  }
}

public static class rndExtension
{
  public static double NextDouble(this Random rng, double minimum, double maximum)
  {
    return rng.NextDouble() * (maximum - minimum) + minimum;
  }
}
```