Engenharia de Computação

Computação Evolucionista

Relatório 1

Sum of different powers

Professor: Samuel Costa Alves Basilio.
Relator: Lorenzo Jordani Bertozzi Luz.

Leopoldina, MG

Entrega: 30/10/2024

I Introdução.

Neste estudo, trabalhamos com a Função de Soma de Diferentes Potências, uma função conhecida na otimização matemática devido à sua característica unimodal. O gráfico demonstrativo desta função é caracterizado por um paraboloide circula representado na "Figura 1", e a função é definida pelo somatório exemplificado na equação (1).

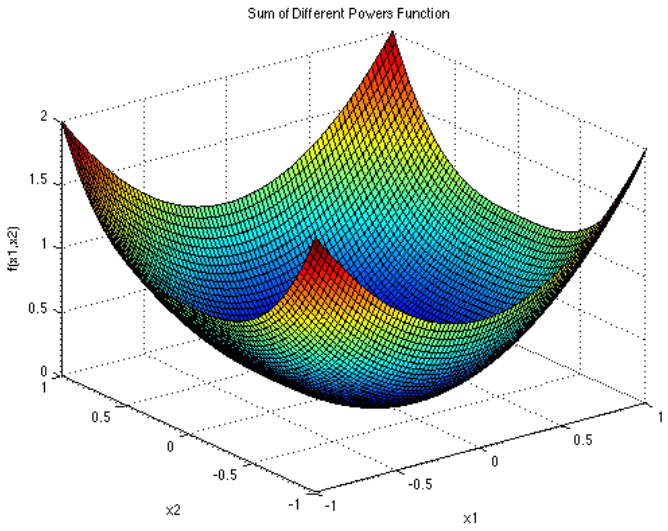


Figura 1: exemplo de plotagem da função de soma de deferente potenciações.

$$f(\mathbf{x}) = \sum_{i=1}^{d} |x_i|^{i+1} \tag{1}$$

Nesta função, **d** representa a dimensão do vetor **x**. A função é avaliada no hipercubo $x_i \in [-1, 1]$ para $i = 1, \ldots, d$, com um mínimo global $f(\mathbf{x}^*) = 0$ em $\mathbf{x}^* = (0, \ldots, 0)$. Esse problema é interessante para a otimização devido à sua simplicidade e à possibilidade de análise eficiente de métodos de busca em um espaço de baixa dimensão.

O objetivo deste projeto foi implementar a função da soma de diferentes potências e realizar duas abordagens de busca para encontrar o mínimo: uma busca sequencial e uma busca aleatória, ambas restritas ao domínio especificado. Em seguida, as avaliações dessas abordagens foram analisadas e visualizadas, a fim de comparar sua eficácia na localização do mínimo global.

II Metodologia.

Para abordar a otimização do problema, inicialmente implementamos a função responsável pela soma de nossas potenciações, conforme apresentado no algoritmo 1. Em seguida, aplicamos dois métodos de busca distintos para analisar o desempenho e comportamento da função em diferentes sistemas de busca. O primeiro método, busca sequencial, examina cada elemento da sequência de forma linear até encontrar o resultado desejado [algoritmo 2]. O segundo método, busca aleatória, seleciona elementos de maneira não-determinística, com a intenção de explorar diferentes partes do espaço de busca para potencialmente melhorar a eficiência [algoritmo 3].

Algoritmo 1: Função soma_diferentes_potencias

```
def soma_diferentes_potencias(x):
    return sum(np.abs(x[i]) ** (i + 1) for i in range(len(x)))
```

Busca Sequencial

Nesta abordagem, o espaço de busca [1,1] foi discretizado em 1000 pontos uniformemente espaçados para avaliar a função de forma sequencial. Essa estratégia é adequada para espaços de baixa dimensão, uma vez que permite examinar sistematicamente o espaço de busca e identificar o mínimo.

Algoritmo 2: Busca_Sequencial

```
def busca_sequencial(d, avaliacoes):
    valores = []
    pontos = []
    start_time = time.time()

    for _ in range(avaliacoes):
        # Gera um vetor aleat rio dentro do intervalo [-1, 1]
           para cada dimens o
        x = np.random.uniform(-1, 1, d)

        # Avalia a fun   o soma_diferentes_potencias no ponto
           x
        valor = soma_diferentes_potencias(x)
        valores.append(valor)
        pontos.append((x, valor))

    # Calcula o tempo de execu   o
    tempo_execucao = time.time() - start_time

    # Calcula e retorna as estat sticas
    return {
```

```
        'melhor': min(valores),
        'pior': max(valores),
        'media': np.mean(valores),
        'mediana': np.median(valores),
        'desvio_padrao': np.std(valores),
        'tempo': tempo_execucao,
        'pontos': pontos
    }
```

Busca Aleatória

Nesta abordagem, realizamos 10 execuções independentes, onde cada execução gerou 1000 pontos aleatórios dentro do intervalo [1,1]. Em cada execução, a função foi avaliada em cada ponto gerado. Essa técnica permite uma exploração mais ampla e não estruturada do espaço, captando potenciais mínimos em regiões menos esperadas.

Algoritmo 3: Busca_Aleatória

```
def busca_aleatoria(d, avaliacoes):
    resultados = []
    for _ in range(10):  # 10 execucoes independentes
        inicio = time.time()
        valores = []
        for _ in range(avaliacoes):
            x = np.random.uniform(limite_inf, limite_sup, d)
            valor = soma_diferentes_potencias(x)
            valores.append(valor)
        tempo_execucao = time.time() - inicio
        resultados.append({
            'melhor': min(valores),
            'pior': max(valores),
            'media': np.mean(valores),
            'mediana': np.median(valores),
            'desvio_padrao': np.std(valores),
            'tempo': tempo_execucao
        })
    return pd.DataFrame(resultados)
```

III Métricas e Análise Estatística.

Após a execução de cada abordagem, calculamos as seguintes métricas estatísticas para avaliar a qualidade dos resultados:

- Melhor Resultado: O menor valor de f(x) encontrado.
- Pior Resultado: O maior valor de f(x) encontrado.
- Média e Mediana dos valores obtidos, para entender a tendência central.
- Desvio Padrão dos valores, indicando a variabilidade.
- Tempo de Execução de cada abordagem para avaliar a eficiência.

Ao analisar os dados retornados por cada um dos métodos de busca apresentados na Tabela 1, podemos destacar alguns resultados relevantes. Um exemplo é o valor **0.000032**, evidenciado na tabela, que já na primeira execução se aproxima significativamente do mínimo global, apresentando uma aproximação notavelmente melhor em comparação à busca sequencial, cujo melhor resultado foi **0.000802**. A partir da leitura da tabela, observamos também que todos os piores resultados se aproximam do valor máximo delimitado. Além disso, a média, a mediana e o desvio padrão em ambas as execuções permanecem muito próximos entre si, indicando uma consistência nos resultados obtidos.

Tabela 1: Tempos, velocidades, quantidades de movimento e energias cinéticas antes e depois do choque.

Exct.	Método	M. Result.	P. Result.	Média	Mediana	Des. Padrão	Tempo (s)
0	Aleatória 1	**0.000032**	1.987852	0.830303	0.823264	0.413810	0.167089
1	Aleatória 2	0.004758	1.986282	0.835616	0.825118	0.418093	0.249171
2	Aleatória 3	0.003394	1.981751	0.828679	0.819183	0.416077	0.213977
3	Aleatória 4	0.000851	1.967799	0.837403	0.830100	0.410394	0.087936
4	Aleatória 5	0.001545	1.988818	0.839105	0.833075	0.416778	0.081648
5	Aleatória 6	0.004149	1.989182	0.833446	0.824190	0.415491	0.082732
6	Aleatória 7	0.000292	1.987986	0.830655	0.821258	0.412791	0.115494
7	Aleatória 8	0.003654	1.975575	0.828647	0.817680	0.411342	0.089378
8	Aleatória 9	0.000811	1.990152	0.835524	0.825696	0.416902	0.093232
9	Aleatória 10	0.001182	1.979760	0.828607	0.822418	0.414876	0.097899
10	Sequencial	**0.000802**	1.991194	0.837366	0.832167	**0.416223**	0.051813

IV Resultados.

Para um melhor entendimento dos dados coletados, foram gerados dois tipos de gráficos de dispersão para facilitar a visualização. O primeiro é um gráfico bidimensional (2D), representado na Figura 2, onde é possível observar a variação dos pontos encontrados pela busca sequencial, com destaque para o melhor resultado identificado. O segundo consiste

em gráficos tridimensionais (3D), representados na Figura 3, que ilustram a formação da "onda"de crescimento das operações de soma de potências. Observa-se, a partir da visualização dos gráficos 3D, que os pontos encontrados se aproximam bastante do nosso objetivo inicial, conforme discutido na introdução deste relatório.

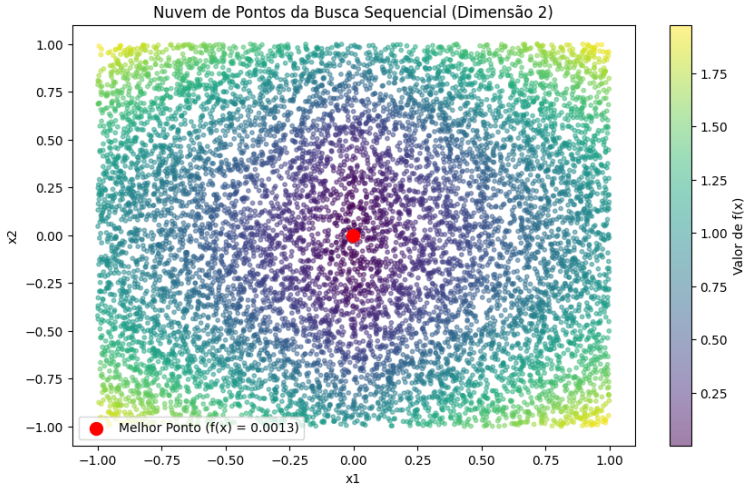
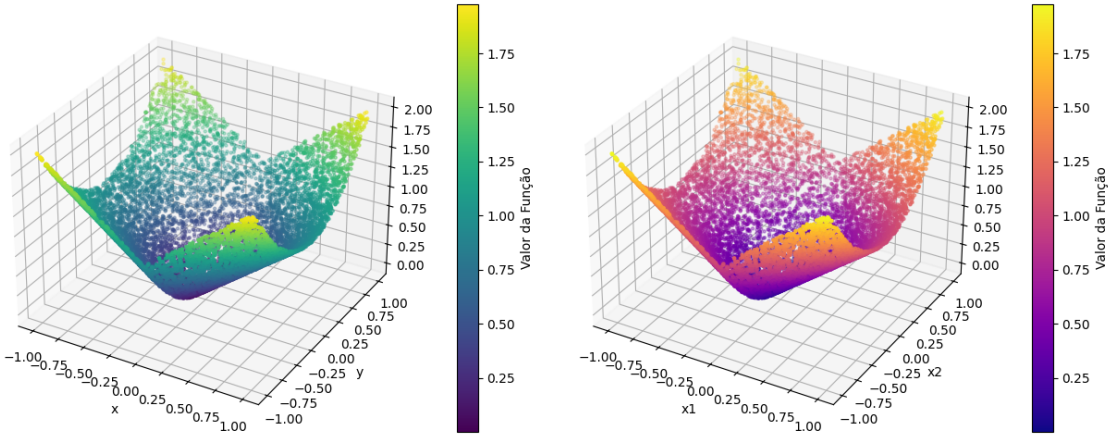


Figura 2: Plotagem Bidimensional da Busca Sequencia



(a) Busca Sequencial (b) Busca Aleatória

Figura 3: Plotagem Tridimensional

Os resultados indicaram uma variação significativa na busca aleatória, evidenciada pelo desvio padrão, bem como pelos melhores e piores valores obtidos nas execuções. A busca sequencial, apesar de seu caráter sistemático, apresentou um valor de função constante, refletindo as limitações da discretização ao explorar o espaço de busca.

V Conclusão.

A análise das duas abordagens sugere que, para funções simples e bem definidas, como a Função de Soma de Diferentes Potências, a busca sequencial pode ser uma estratégia eficiente, embora limitada pela discretização imposta ao espaço de busca. Em contraste, a busca aleatória permite uma exploração mais ampla do espaço, sendo útil para identificar valores próximos ao mínimo, apesar de apresentar uma maior variabilidade nos resultados. Este estudo evidencia a importância de adaptar a abordagem de busca de acordo com a complexidade e a dimensionalidade da função alvo. Ele destaca que métodos estocásticos, como a busca aleatória, podem ser alternativas viáveis em problemas onde a otimização exata não é essencial ou se torna impraticável, proporcionando soluções que balanceiam eficiência e abrangência exploratória.

VI Referências

[1] Sum of different powers function. ([s.d.]). Sfu.ca. Recuperado 6 de novembro de 2024, de https://www.sfu.ca/ ssurjano/sumpow.html