Computação Quântica pela Álgebra Linear: Ortonormalização

Autora

Maria Clara Sales

1 Introdução

A computação quântica ainda é um campo novo e complexo, porém possui uma base grande na álgebra linear. Devido a isso, operações quânticas podem ser representadas por operações básicas, tornando o compreendimento delas mais tátil. O objetivo desse trabalho é representar conceitos e operações que normalmente usariam de vetores complexos para obter um resultado de uma forma simples, apenas utilizando álgebra linear básica. Além de apresentar uma forma de ortonormalização de espaços quânticos, importante para cálculos de circuitos quânticos. As operações são realizadas através da linguagem Python e da biblioteca NumPy e suas representações em gráfico são obtidas pela biblioteca MatlplotLib.

2 Álgebra Linear na Quântica

A mecânica quântica (um dos pilares da computação quântica) se baseia na álgebra linear para seus cálculos e representações. Entre eles, podemos afirmar que um estado quântico é um vetor binário de n-dimensões. A partir disso, todas as operações são desdobramentos das operações vetoriais: projeções, combinações lineares, produto interno, entre outros.

Utilizaremos nesse projeto essa interpretação.

3 Operações

As operações trabalhadas no projeto são de normalização, criação de sobreposições, medição quântica e ortonormalização.

3.1 Normalização

A normalização de vetores é uma forma de ajuste de um vetor qualquer a um tamanho específico, geralmente para o comprimento um, e é dita pela divisão do próprio vetor pela sua norma.
Dado um vetor $\vec{v} = (v_1, v_2, \ldots, v_n)$, sua forma normalizada é dada por:

$$\vec{u} = \frac{\vec{v}}{\|\vec{v}\|}$$

onde $\|\vec{v}\|$ é a **norma** do vetor $\vec{v}$, usaremos a norma euclidiana:

$$\|\vec{v}\| = \sqrt{v_1^2 + v_2^2 + \ldots + v_n^2}$$

Em vetores que representam estados quânticos, essa abordagem não é modificada, as mesmas operações são utilizadas. A normalização facilita o processamento desses estados, além de auxiliar outros algoritmos quânticos como HHL ou Algoritmo de Grover. Eis um exemplo:

Dado o vetor que representa um estado de sobreposição $\vec{v} = [1.0, 1.0, 0.0]$, iremos calcular sua normalização $\vec{u}$.

Passo 1: A norma Euclidiana $\|\vec{v}\|$ é dada por:

$$\|\vec{v}\| = \sqrt{v_1^2 + v_2^2 + v_3^2}$$

Substituímos em $\vec{v}$:

$$\|\vec{v}\| = \sqrt{(1.0)^2 + (1.0)^2 + (0.0)^2} = \sqrt{1.0 + 1.0 + 0.0} = \sqrt{2.0}$$

Passo 2: A normalização é obtida dividindo cada componente de $\vec{v}$ pela norma $\|\vec{v}\|$:

$$\vec{u} = \frac{\vec{v}}{\|\vec{v}\|}$$

Substituímos os valores:

$$\vec{u} = \left[\frac{1.0}{\sqrt{2.0}}, \frac{1.0}{\sqrt{2.0}}, \frac{0.0}{\sqrt{2.0}}\right]$$

Simplificando cada componente, temos:

$$\vec{u} = \left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0\right]$$

3.1.1 Código

```
1
2 def norma(v):
3     """ norma euclidiana """
4     return np.sqrt(np.sum(v**2)) # realiza a raiz do somatorio de
      todos os elementos do vetor ao quadrado
5
6 def normalizar(v):
7     """"Operacao de normalizar"""
8     return v / norma(v)
```

3.2 Combinações lineares e sobreposições quânticas

Combinações lineares são relações entre vetores linearmente independentes u e v (que não são combinações lineares entre si) e através dessas podemos criar novos vetores apenas somando-os e/ou multiplicando-os por constantes:

$$w = C_0.u + C_1.v$$

Seguindo a mesma lógica da combinação linear, há dois estados independentes $|1\rangle$ e $|0\rangle$ em computação quântica. Esses podem ser representados por vetores como $1 = [0, 1]$ e $0 = [1, 0]$ ou em outras dimensões, contanto que siga a lógica da independência linear. O bit quântico é uma "combinação" dos estados quânticos, os quais possuem um peso na probabilidade da binaridade do bit.

A sobreposição quântica ocorre quando algo (o bit, nesse caso) está em múltiplos estados ao mesmo tempo e com "pesos" diferentes. Na computação, a sobreposição quântica é o estado de um bit e também o quanto um bit está propenso a ser 0 ou 1, atribuindo probabilidade aos estados $|1\rangle$ e $|0\rangle$. Dessa forma, é possível representar a sobreposição do bit quântico como uma combinação linear dos estados, onde os coeficientes são as probabilidades da leitura digital do bit ($C_0 + C_1 = 1$). Essa representação pode ser usada em cálculos para um manuseio mais fácil dessa sobreposição.

$$q = C_0.[1, 0] + C_1.[0, 1]$$

3.2.1 Código

```
# Vetores base
v1 = np.array([1.0, 0.0]) # |0>
v2 = np.array([0.0, 1.0]) # |1>

# Combinacao linear (sobreposicao)
c1, c2 = 0.5, 0.5  # Coeficientes
q = c1 * v1 + c2 * v2


# Normalizando
q_normalizado = q / np.linalg.norm(q)

print("Estado em superposicao:", q)
```

3.3 Medição quântica

Quando há a necessidade de comparar dois vetores, muitas vezes a projeção é usada. Projeções ortogonais, por exemplo, podem ser utilizadas quando é necessário analisar um sistema tridimensional em um subespaço bidimensional. A projeção ortogonal se dá pelo mapeamento de vetores em subespaços, minimizando as distâncias perpendiculares e de forma que a diferença entre o vetor original e o projetado seja ortogonal ao subespaço. De forma simples, decompõe o vetor em um espaço de forma ortogonal.

Na computação quântica, a leitura digital segue um conceito semelhante. A medição quântica é o processo de colapsar o estado de sobreposição de um qubit (ou bit quântico) em um dos estados da base computacional do espaço quântico utilizado, resultando em um valor discreto (0 ou 1, no caso de um único qubit). Quando o espaço quântico de um qubit utilizado é ortonormal, a projeção utilizada é a ortogonal uma vez que as bases são ortogonais entre si. Assim, é possível definir o valor de um qubit projetando-o nas bases e calculando a probabilidade. Essa probabilidade é dada pelo *quadrado do módulo do produto interno entre o estado do sistema e o estado da base de medição.*

4 Algoritmo de Gram-Schmidt e ortonormalidade

A ortonormalidade é fundamental na computação quântica. Utilizando conjunto de vetores ortornormais, cálculos de probabilidades (utilização de projeções ortogonais para medição quântica) e resolução de equações se tornam mais simples, pois, ao garantir a ortonormalidade, várias expansões podem ser utilizadas.

4.1 Gram-Schmidt

Um dos métodos que podem ser usados é o de Gram-Schmidt que, além de gerar espaços/conjuntos ortonormais, pode ajudar na redução de dimensões de espaços quânticas encontrando bases ortonormais menores que não influenciaram no conteúdo do espaço original.

O método de Gram-Schmidt é um processo que transforma um conjunto de vetores linearmente independentes em um conjunto ortonormal. Também é utilizado para formar espaços vetoriais ortonormais, que é onde a computação quântica se faz presente. O processo primeiro remove as projeções de cada vetor nos vetores anteriores para eliminar as componentes dependentes de outros vetores. Em seguida, normaliza os vetores resultantes. O objetivo é gerar um conjunto de vetores que são ortogonais entre si e de norma unitária, mantendo o subespaço original. As funções feitas nesse projeto são para auxiliar o algoritmo implementado de tal forma:

```
    def gram_schmidt(vetores):
    """
    Realiza o processo de Gram-Schmidt para ortonormalizar uma
    lista de vetores.

    Parameters:
    vectors: Lista de vetores (arrays numpy).  .

    Returns:
    Ortonormaliza os vetores e retorna uma nova lista de vetores
    ortonormais.
    """
    ortho_vetores = [] # Conjunto de vetores ortonormais
    for v in vetores:
        for u in ortho_vetores:
            v -= np.dot(u, v) * u # Retirando todas as possiveis
    projecoes
        if norma(v) > 1e-10: # Eliminando valores nulos
            ortho_vetores.append(normalizar(v)) # Normalizando
    return ortho_vetores

# Criar uma base ortonormal
base_ortonormal = gram_schmidt(vetores)

np.set_printoptions(precision=5) # Precisao de 5 algarismos
# Exibir os vetores da base
for i, vec in enumerate(base_ortonormal):
    print(f"Vetores ortonormalizados {i+1}: {vec}")
```

4.2 Aplicação em estados quânticos

Os vetores utilizados para representar os estados quânticos é Um espaço de Hilbert é um espaço vetorial completo com produto interno (onde vetores podem ser somados e multiplicados por escalares, além de possuírem uma métrica que permite calcular distâncias e ângulos). A "completude" do espaço significa que não há lacunas, garantindo que todas as sequências convergentes dentro do espaço têm limites bem definidos, o que o torna vantajoso para a computação e mecânica quântica. Algumas regras desse espaço são:

1. O vetor pode ser uma lista infinita ou finita, de números reais ou complexos (principalmente usado em vetores complexos e quânticos);
2. O produto interno deve ser linear, simétrico e positivo definido;
3. Os vetores de estado geralmente têm norma unitária, devido a lógica de probabilidade de medição.

4.2.1 Aplicação prática

4.3 Primeiro Input

Entrada usada:

```
    vetores_1 = [
    np.array([1.0, 1.0, 0.0]),
    np.array([1.0, 0.0, 1.0]),
    np.array([0.0, 1.0, 1.0])
]
```

Saída obtida:

```
Vetores ortonormalizados 1: [0.70711 0.70711 0.00000]
Vetores ortonormalizados 2: [ 0.40825 -0.40825  0.81650]
Vetores ortonormalizados 3: [-0.57735  0.57735  0.57735]
```

É possível notar que a saída é imprecisa, mas o motivo é a simplicidade do algoritmo para esse projeto. Mesmo simples, ele consegue gerar um conjunto ortonormal sem problemas maiores.

```
Propriedades do espaco quantico 1:
Norma do vetor 1: 0.9999999999999999
Produto interno entre vetor 1 e vetor 2: 1.1102230246251565e-16
Produto interno entre vetor 1 e vetor 3: 5.551115123125783e-17
Norma do vetor 2: 1.0
Produto interno entre vetor 2 e vetor 1: 1.1102230246251565e-16
Produto interno entre vetor 2 e vetor 3: -1.1102230246251565e-16
Norma do vetor 3: 0.9999999999999999
Produto interno entre vetor 3 e vetor 1: 5.551115123125783e-17
Produto interno entre vetor 3 e vetor 2: -1.1102230246251565e-16
```

4.4 Input 2

Entrada usada:

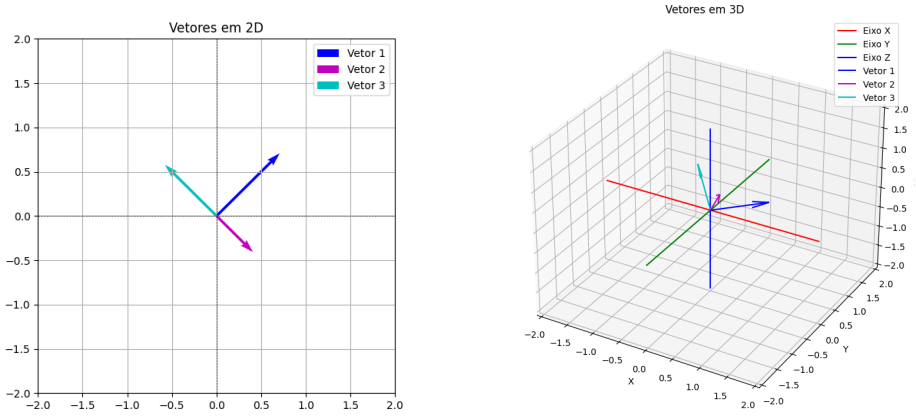
```
    vetores_2 = [
    np.array([1.0, 2.0, 3.0]),
    np.array([4.0, 5.0, 6.0]),
    np.array([7.0, 8.0, 9.0]),
    np.array([1.0, 1.0, 1.0])]
```

Saída obtida:

```
Vetores ortonormalizados 1: [0.26726 0.53452 0.80178]
Vetores ortonormalizados 2: [ 0.87287  0.21822 -0.43644]
```

Verificação:

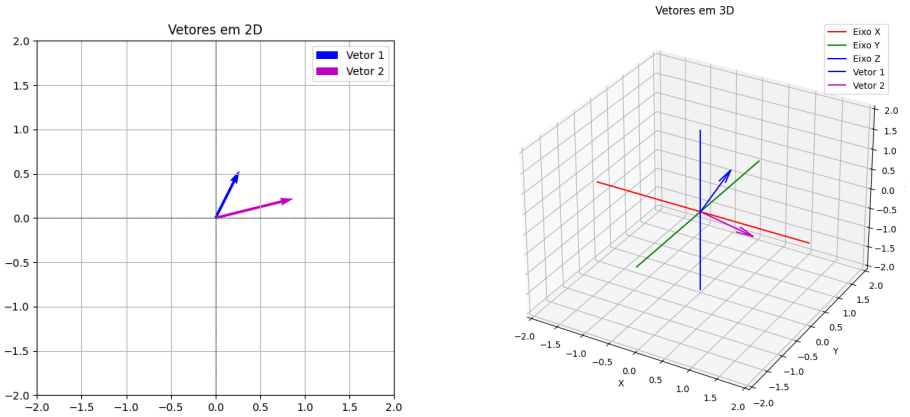
```
Propriedades do espaco quantico 1:
Norma do vetor 1: 1.0
Produto interno entre vetor 1 e vetor 2: -3.885780586188048e-16
Norma do vetor 2: 1.0
Produto interno entre vetor 2 e vetor 1: -3.885780586188048e-16
```



(a) Input 1 no 2D

(b) Input 1 no 3D

Figure 1: Comparação entre representações em 2D e 3D para o Input 1.



(a) Input 2 no 2D

(b) Input 2 no 3D

Figure 2: Comparação entre representações em 2D e 3D para o Input 2.

5 Aplicações na realidade e conclusão

A ortonormalização e outras teorias da álgebra linear são importantes em diversas vertentes da computação quântica. Entre elas, temos:

1. Emanharamento e entrelaçamento quântico
2. Processamento do Teletransporte quântico
3. Métodos numéricos e análises com algoritmos quânticos
4. Análise e simulação de circuitos e sistemas quânticos

A álgebra linear, além de auxiliar a computação quântica em si, auxilia na visualização e na compreensão da mecânica quântica de forma simplificada. A partir de simples operações, é possível explicar noções de quântica sem a utilização de vetores complexos e quânticos, ou até utilizando os próprios qubits.

6 Referências

TANG, Richard J. Quantum Algorithms via Linear Algebra. Cambridge: Cambridge University Press, 2014.

CHILD, Andrew M.; LUONG, Daniel. Quantum Linear Systems Algorithms: A Primer. New York: Springer, 2022.

GRAVES, Lawrence M. Introduction to Hilbert Spaces with Applications. 2. ed. New York: Wiley, 1985.

NEARING, Francis P. Quantum Mechanics: Concepts and Applications. 3. ed. Cambridge: Cambridge University Press, 2018.

STRANG, Gilbert. Linear Algebra and Its Applications. 4. ed. Belmont: Brooks/Cole, 2006

RECOGNA NLP. Bases matemáticas da computação quântica: álgebra linear, superposição e entrelaçamento. Medium, 10 fev. 2023. Disponível em: https://medium.com/@recogna.nlp/bases-matem