A Otimização Linear no Problema da Dieta

Davi Juliano Ferreira Alves
RA: 133595

Pedro Diniz Sakai
RA: 133696

Resumo

Otimização Linear é um ramo das otimizações que visa trabalhar somente com grandezas e variáveis lineares. Devido a essa ferramenta, há uma certa aplicação das soluções e métodos matemáticos nos problemas dados como reais. Para esse trabalho, foram enunciados três métodos para solucionar Problemas de Otimização Linear: O Método do Gráfico, o Método Simplex e o PuLP. O problema, pelo qual estivemos desvendando uma solução, foi o problema da dieta, problema muito conhecido pelos estudiosos de Otimização, que visa solucionar uma dieta balanceada e otimizada para um menor custo.

Palavras-chave: Programação Linear, Problema, Dieta, Problemas Reais, PuLP.

1 Introdução

A Otimização Linear (ou Programação Linear) é um ramo da Pesquisa Operacional que visa resolver problemas lineares aplicando os métodos de restrição para maximizar ou minimizar a função objetivo.

A Otimização Linear foi criada, de acordo com [4], por um grupo de cientistas que foram convocados na Inglaterra para estudar problemas de estratégia e de tática associados com a defesa do país. O objetivo era decidir sobre a utilização mais eficaz de recursos militares limitados. A convocação deste grupo marcou a primeira atividade formal de Pesquisa Operacional. Os resultados positivos conseguidos pela equipe de Pesquisa Operacional (PO) inglesa motivaram os Estados Unidos a iniciarem atividades semelhantes. Apesar de ser creditada à Inglaterra a origem da Pesquisa Operacional, sua propagação deve-se, principalmente, à equipe de cientistas liderada por George B. Dantzig, dos Estados Unidos, convocada durante a Segunda Guerra Mundial. Ao resultado deste esforço de pesquisa, concluído em 1947, deu-se o nome de Método Simplex.

Primeiramente, o método mais rústico e simples de Otimização Linear é o Método do Gráfico. De acordo com [5], o método do Gráfico é uma técnica de solução utilizada para ilustrar problemas de duas ou três variáveis de decisão, visto que mais de três variáveis não consegue-se ilustrar. Esta técnica permite a visualização da resolução de problemas de programação linear.

Entre os métodos de Otimização Linear, temos o Método Simplex, que é um dos métodos mais usados em Otimização Linear. De acordo com [3], o algoritmo Simplex é o método mais utilizado para a solução de problemas de programação linear. É um procedimento algébrico iterativo que parte de uma solução básica factível inicial e busca, a cada iteração, uma nova solução básica factível com melhor valor na função objetivo, até que o valor ótimo seja atingido.

Além dos Métodos de Otimização Linear, utilizamos também um solver computacional da linguagem Python: o PuLP. De acordo com a documentação do PuLP, [2], temos que o PuLP é dividido em 5 etapas:

- Obter a descrição do problema
- Formular matematicamente o problema

- Resolver matematicamente o problema
- Performar algumas análises que podem ser ideais
- Apresentar a análise e o resultado ótimo

Além dessas etapas, temos a generalização dos conceitos no esquema dado:

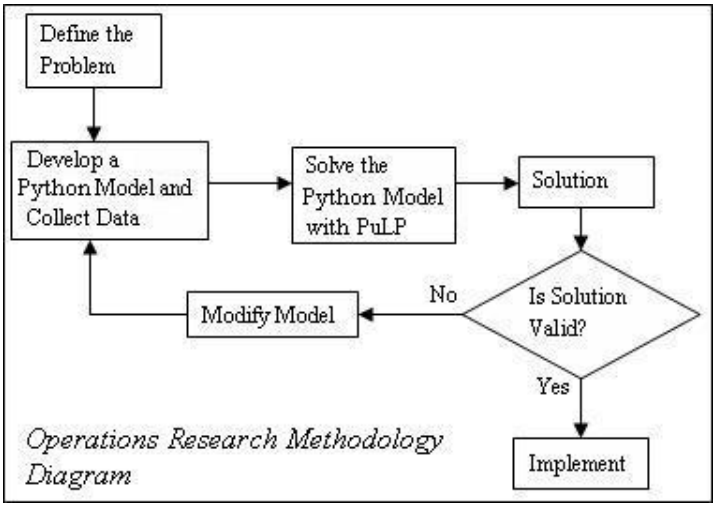

Figura 1: Diagrama de Metodologia da Pesquisa Operacional

2 Fundamentação Teórica

Na resolução de um PPL, procura-se determinar a solução ótima, que deve satisfazer aos seguintes teoremas:

Teorema 1 *Se o PPL tem solução ótima, então esta solução encontra-se em um dos pontos extremos do poliedro convexo (região de soluções viáveis).*

Teorema 2 *Se o PPL tem soluções viáveis e não vazias, então existe uma solução ótima.*

Teorema 3 *Um conjunto de soluções viáveis de um PPL é um conjunto convexo.*

Teorema 4 *O número de soluções viáveis de um PPL tem um número finito de pontos extremos (vértices).*

Os teoremas estão enunciados no [5].

2.1 Método do Gráfico

Sabemos que a representação gráfica de uma equação linear com duas variáveis é uma reta ($y = ax + b$). E a representação gráfica de uma inequação linear com duas variáveis é um dos semiplanos definidos pela reta correspondente à equação. Aplicaremos isso em um problema do método do gráfico:

Exemplo: Considere o PPL abaixo

$$\text{MAX: } Z = 4x_1 + 2x_2.$$
$$\text{Sujeito à: } \begin{cases} 4x_1 + 1x_2 \leq 30, \\ 0.33x_1 + 0.66x_2 \leq 8. \end{cases}$$

Utilizando o Método do gráfico, temos os seguintes semiplanos:

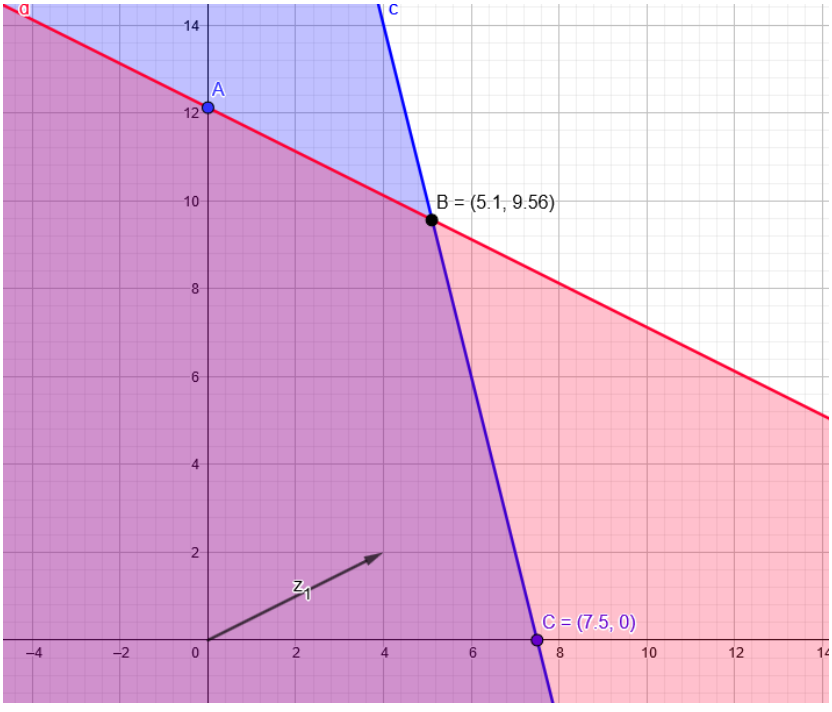

Figura 2: Semi-Planos das Restrições, os Pontos Notáveis e o ∇Z

Analisando agora os pontos notáveis (7.5,0), (5.1,9.56) e (0,12.12), temos:

$$\begin{aligned} Z(7.5, 0) &= 24.24, \\ Z(5.1, 9.56) &= 39.52, \\ Z(0, 12.12) &= 30. \end{aligned}$$

Portanto, pelo método do gráfico, temos que para $x_1 = 5.1$ e $x_2 = 9.56$, a função tem seu máximo $Z(5.1, 9.56) = 39.52$.

2.2 Método Simplex

O método Simplex é uma técnica iterativa que busca a melhor solução da função objetivo em cada passo, otimizando a solução final. O método parte de uma solução básica viável e segue realizando deslocamentos pelas arestas do polígono encontrado pelo Método Gráfico, ou seja, partimos do vértice atual até um adjacente que melhore o valor da função objetivo. Sempre que exista região viável, e como seu número de vértices e de arestas é finito, será possível encontrar a solução.

Para elucidar o método, mostraremos um exemplo passo a passo. Suponha que desejamos maximizar a seguinte função objetivo, sujeito às seguintes restrições

$$\begin{aligned} \text{MAX: } & z = 3x_1 + 2x_2. \\ \text{Sujeito à: } & 2x_1 + x_2 \leq 18, \\ & 2x_1 + 3x_2 \leq 42, \\ & 3x_1 + x_2 \leq 24, \\ & x_1,\, x_2 \geq 0. \end{aligned}$$

Passo 1: Primeiramente, transformamos o modelo matemático acima na forma padrão do simplex. Para isso, convertemos as desigualades em igualdades adicionando uma variável de folga x_i para cada restrição. Em seguida, igualamos a função objetivo à zero. Dessa forma, obtemos

$$\begin{aligned} \text{MAX: } & z - 3x_1 - 2x_2 = 0. \\ \text{Sujeito à: } & 2x_1 + x_2 + x_3 = 18, \\ & 2x_1 + 3x_2 + x_4 = 42, \\ & 3x_1 + x_2 + x_5 = 24, \\ & x_1,\, x_2 \geq 0. \end{aligned}$$

Passo 2: Agora, transferimos a forma padrão acima para uma tabela inicial, esta tabela contém os coeficientes das variáveis de decisão e das variáveis de folga e as limitações em suas colunas, e os coeficientes das variáveis de folga e z nas linhas.

	x_1	x_2	x_3	x_4	x_5	l
x_3	2	1	1	0	0	18
x_4	2	3	0	1	0	42
x_5	3	1	0	0	1	24
z	-3	-2	0	0	0	0

Esta solução inicial retorna $z = 0$. Observe que a $1^{\underline{a}}$ iteração corresponde a solução da origem no Método Gráfico.

Passo 3: A seguir realizaremos as iterações do método atentando-se ao critério de parada. Podemos estar interessados tanto em maximizar, como no exemplo, quanto em minimizar. Em cada uma das opções existe o critério de parada, a condição de entrada na base e a condição de saída da base.

• **Maximizar:**

-Critério de parada: os valores da linha z são não-negativos.

-Condição de entrada na base: o menor valor da linha z indica a variável x_j que entra na base.

-Condição de saída da base: depois de obter a variável de entrada, determina-se a variável de saída por meio do menor quociente l/x_j dos valores estritamente positivos.

• **Minimizar:**

-Critério de parada: os valores da linha z são não-positivos.

-Condição de entrada na base: o maior valor da linha z indica a variável x_j que entra na base.

-Condição de saída da base: depois de obter a variável de entrada, determina-se a variável de saída por meio do menor quociente l/x_j dos valores estritamente negativos.

Chamaremos o elemento que está na coluna da variável que sai da base e na linha da variável que entra na base de pivô.

	x_1	x_2	x_3	x_4	x_5	l
x_3	2	1	1	0	0	18
x_4	2	3	0	1	0	42
x_5	3	1	0	0	1	24
z	-3	-2	0	0	0	0

Na tabela do exemplo, x_1 é a variável que entra na base, pois dentre os valores da linha z, -3 é o menor. Em seguida, temos os seguintes quocientes l/x_1: $24/3 = 8$, $42/2 = 21$ e $18/2 = 9$. Logo, a variável que sai da base é x_5 e o pivô é 3.

Passo 4: Agora, atualizamos a tabela. Primeiramente dividimos as entradas da linha do pivô pelo pivô:

	x_1	x_2	x_3	x_4	x_5	l
x_3	2	1	1	0	0	18
x_4	2	3	0	1	0	42
x_1	1	1/3	0	0	1/3	8
z	-3	-2	0	0	0	0

Em seguida, zeramos as entradas da coluna do pivô. Seja L_{x_j} a linha do pivô, L uma linha e a uma constante, realizaremos a operação $aL_{x_j} + L_{x_i}$, para cada linha L que não a do pivô. No exemplo, devemos realizar as operações $-2L_{x_1} + L_{x_4}$, $-2L_{x_1} + L_{x_3}$ e $3L_{x_1} + z$, obtendo

	x_1	x_2	x_3	x_4	x_5	l
x_3	0	1/3	1	0	-2/3	2
x_4	0	7/3	0	1	-2/3	26
x_1	1	1/3	0	0	1/3	8
z	0	-1	0	0	1	24

Passo 5: Agora, verificamos o critério de parada. Caso seja cumprido, paramos. Caso contrário, seguimos para a próxima iteração. No exemplo, devemos continuar pois ainda existem valores negativos na linha de z. A próxima iteração mostra que -1 é o menor valor da linha de z, e portanto, x_2 entra na base,

	x_1	x_2	x_3	x_4	x_5	l
x_2	0	1/3	1	0	-2/3	2
x_4	0	7/3	0	1	-2/3	26
x_1	1	1/3	0	0	1/3	8
z	0	-1	0	0	1	24

Já o menor quociente l/x_2 estritamente positivo é 6, que corresponde a linha x_3 que portanto sai da base e 1/3 é o pivô. Logo,

	x_1	x_2	x_3	x_4	x_5	l
x_2	0	1	3	0	-2	6
x_4	0	7/3	0	1	-2/3	26
x_1	1	1/3	0	0	1/3	8
z	0	-1	0	0	1	24

e

	x_1	x_2	x_3	x_4	x_5	l
x_2	0	1	3	0	-2	6
x_4	0	0	-7	1	4	12
x_1	1	0	-1	0	1	6
z	0	0	3	0	-1	30

Passo 6: Verificamos o critério de parada. Contiuamos, no exemplo, pois ainda existem valores negativos na linha de z. A próxima iteração mostra que -1 é o menor valor da linha de z, e portanto, x_5 entra na base,

	x_1	x_2	x_3	x_4	x_5	l
x_2	0	1	3	0	-2	6
x_4	0	0	-7	1	4	12
x_1	1	0	-1	0	1	6
z	0	0	3	0	-1	30

Já o menor quociente l/x_5 estritamente positivo é 3, que corresponde a linha x_4 que portanto sai da base e 4 é o pivô.

	x_1	x_2	x_3	x_4	x_5	l
x_2	0	1	3	0	-2	6
x_5	0	0	-7/4	1/4	1	3
x_1	1	0	-1	0	1	6
z	0	0	3	0	-1	30

e

	x_1	x_2	x_3	x_4	x_5	l
x_2	0	1	-1/2	1/2	0	12
x_5	0	0	-7/4	1/4	1	3
x_1	1	0	3/4	-1/4	0	3
z	0	0	5/4	1/4	0	33

Passo 7: Quando o critério de parada é satisfeito, a solução ótima são as entradas de cada linha com a coluna l. No exemplo, temos que as entradas da linha z são todas não-negativas, e portanto, o critério de parada é satisfeito. A solução ótima é $x_1 = 3$, $x_2 = 12$, $x_3 = 0$, $x_4 = 0$, $x_5 = 3$, que retorna $z = 33$.

2.3 Solver PuLP(Python)

O solver PuLP do Python, de acordo com [2], resolve os problemas juntamente com passos específicos de modelagem matemática:

- Identificar as variáveis de decisão com atenção especial às unidades.

- Formulando a função objetivo usando as variáveis de decisão, pode-se construir uma função objetivo para minimizar ou para maximizar. A função objetivo normalmente reflete o custo total, ou lucro total, para um determinado valor das variáveis de decisão.

- Formular as restrições, tanto lógicas quanto explícitas, para a descrição do problema. Novamente, as restrições são expressas em termos de variáveis de decisão.

- Identificar os dados necessários para a função objetivo e restrições. Para resolver seu programa matemático, você precisará ter alguns "números rígidos" como limites de variáveis e/ou coeficientes variáveis em sua função objetivo e/ou restrições.

De acordo com [2], o PuLP utiliza o Método dos Pontos Interiores e o Método Simplex para a solução ótima de problemas relativamente pequenos. Para problemas de maior complexidade e de dados muito longos, o PuLP utiliza heurísticas para solucionar o problema de maneira rápida e com aproximações relativamente interessantes da solução ótima, visto que uma heurística não garante solução ótima.

Para melhor entendimento do método do Solver PuLP, aplicaremos em um exemplo:

Exemplo: Considere o PPL abaixo

$$\begin{aligned} \text{MIN:} &= 0.26x_1 + 0.32x_2. \\ \text{Sujeito à:} &= \begin{cases} 0.07x_1 + 0.21x_2 \geq 0.34, \\ 0.82x_1 + 0.79x_2 \geq 2.64. \end{cases} \end{aligned}$$

Colocando e modelando o problema no código fonte do Python 3.3, temos:

```
import numpy as np
import pulp

# Definindo o problema como de maximizacao
prob = pulp.LpProblem('Exercicio', pulp.LpMinimize)

# Definindo as variaveis de decisao
x1 = pulp.LpVariable('x1', lowBound=0, cat='Continuous')
x2 = pulp.LpVariable('x2', lowBound=0, cat='Continuous')

#Definindo a funcao a ser minimizada
MIN=0.26*x1 + 0.32*x2

#Adicionando a funcao-objetivo
prob += MIN

#Definindo a funcao da reta R
reta_r = 0.07*x1 + 0.21*x2

#Adicionando a funcao da reta R nas restricoes
prob += (reta_r>=0.34)

#Definindo a funcao da reta S
reta_s = 0.82*x1 + 0.79*x2

#Adicionando a funcao da reta S nas restricoes
prob += (reta_s>=2.64)

#escrevendo o problema de otimizacao linear
print(prob)

# Resolvendo o problema
optimization_result = prob.solve()
```

```
34
35 # Verificando se a solucao otima foi encontrada
36 assert optimization_result == pulp.LpStatusOptimal
37
38 #mostrando o resultado
39 for var in (x1, x2):
40     print('Solucao otima em {}: {:1.0f}'.format(var.name, var.value()))
41
42 OUTPUT:
43
44 Exercicio:
45 MINIMIZE
46 0.32*x1 + 0.26*x2 + 0.0
47 SUBJECT TO
48 _C1: 0.21 x1 + 0.07 x2 >= 0.34
49
50 _C2: 0.79 x1 + 0.82 x2 >= 2.64
51
52 VARIABLES
53 x1 Continuous
54 x2 Continuous
55
56 Solucao otima em x1: 2
57 Solucao otima em x2: 1
```

Portanto, o Solver do Python retorna nas variáveis de input uma solução ótima dada pela função objetivo dada, que no caso, o valor ótimo para o $x_1 = 2$ e para $x_2 = 1$.

3 Desenvolvimento

3.1 Problema da Dieta

De acordo com [1], o problema clássico da dieta é organizar uma nutrição saudável e pagar o menor preço por ela. Foi o protótipo para problemas de programação linear, ramo desenvolvido nos anos 1940 e que é agora usado em uma ampla gama de aplicações.

Tem-se, como objetivo, adaptar o problema da dieta, para uma situação real, conforme plano e grupos alimentares elaborados por uma nutricionista, de acordo com a idade, peso e quantidade de calorias que podem ser ingeridas no dia-a-dia de uma pessoa.

Seguindo a tabela de [5], temos uma tabela com alimentos variados que fazem parte da dieta uma certa pessoa, entre os alimentos, temos pão integral, queijo cottage, mamão, nozes, salada crua, feijão, arroz integral, frango grelhado, maçã, tapioca, ovo, atum ralado e iogurte. Para as restrições, temos a quantidade necessária para a manutenção da saúde de quem se alimenta com a menor quantidade de recursos possível, ou seja, minimizando a quantidade de comida a ser consumida. Entre essas restrições, temos energia(kcal), proteína(g), cálcio(g), sódio(g), ferro(g), vitaminas(g), tamanho da porção e por último e o que queremos minimizar, o preço de cada porção.

	Pão Integral	Queijo Cottage	Mamão	Nozes	Salada Crua	Feijão	Arroz Integral	Frango Grelhado	Maçã	Tapioca	Ovo	Atum Ralado	Iogurte
Energia (kcal)	136	85	45	100	25	132	25	165	72	68	77	17	99
Proteína (g)	4,6	17,2	0,8	2,3	1,5	8,8	0,5	31	0,3	0	6,2	3	3,9
Cálcio (g)	0,06	0,03	0,02	0	0,02	0,03	0	0,02	0,01	0	0,03	0	0,14
Sódio (g)	0,276	0,013	0,005	0	0,08	0,001	0	0,074	0,001	0,037	0,139	0,069	0,053
Ferro (g)	0,1	0,01	0,01	0	0,03	0,12	0	0,06	0,01	0,02	0,03	0	0
Vitaminas (g)	0	0,01	0,77	0	0,93	0	0	0	0,12	0	0,06	0	0,02
Tamanho da Porção	52g	100g	100g	15g	100g	100g	20g	100g	140g	20g	50g	20g	100g
Preço (pôr porção)	0,93	5,24	1,29	3,05	0,5	0,63	0,09	0,89	1,95	0,21	0,99	0,61	1,69

Figura 3: Imagem do Problema da Dieta de [5]

Primeiramente, modelaremos matematicamente para logo em seguida, usar o Solver do Python: PuLP. A modelagem segue o seguinte modelo dado por [5]:

Função Objetivo a ser Minimizada: $0,93x_1 + 5,24x_2 + 1,29x_3 + 3,05x_4 + 0,5x_5 + 0,63x_6 + 0,09x_7 + 0,89x_8 + 1,95x_9 + 0,21x_{10} + 0,99x_{11} + 0,61x_{12} + 1,69x_{13}$;

Restrição da Energia: $136x_1 + 85x_2 + 45x_3 + 100x_4 + 25x_5 + 132x_6 + 25x_7 + 165x_8 + 72x_9 + 68x_{10} + 77x_{11} + 17x_{12} + 99x_{13} \geq 1870$;

Restrição da Proteína: $4,6x_1 + 17,2x_2 + 0,8x_3 + 2,3x_4 + 1,5x_5 + 8,8x_6 + 0,5x_7 + 31x_8 + 0,3x_9 + 0x_{10} + 6,2x_{11} + 3x_{12} + 3,9x_{13} \geq 46$;

Restrição do Cálcio: $0,06x_1 + 0,03x_2 + 0,02x_3 + 0x_4 + 0,02x_5 + 0,03x_6 + 0x_7 + 0,02x_8 + 0,01x_9 + 0x_{10} + 0,03x_{11} + 0x_{12} + 0,14x_{13} \geq 1$;

Restrição do Sódio: $0,276x_1 + 0,013x_2 + 0,0055x_3 + 0x_4 + 0,08x_5 + 0,001x_6 + 0x_7 + 0,074x_8 + 0,001x_9 + 0,037x_{10} + 0,139x_{11} + 0,069x_{12} + 0,053x_{13} \geq 1,5$;

Restrição do Ferro: $0,1x_1 + 0,01x_2 + 0,01x_3 + 0x_4 + 0,03x_5 + 0,12x_6 + 0x_7 + 0,06x_8 + 0,01x_9 + 0,02x_{10} + 0,03x_{11} + 0x_{12} + 0x_{13} \geq 0,018$;

Restrição das Vitaminas: $0x_1 + 0,01x_2 + 0,77x_3 + 0x_4 + 0,93x_5 + 0x_6 + 0x_7 + 0x_8 + 0,12x_9 + 0x_{10} + 0,06x_{11} + 0x_{12} + 0,02x_{13} \geq 0,775$;

Restrição da Não-Negatividade: $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13} \geq 0$.

Problema, portanto, é dado por uma pessoa que deseja fazer uma dieta de menor custo possível com uma quantidade mínima de nutrientes. Quanto de cada alimento essa pessoa deve consumir de acordo com a tabela dada? E quanto essa pessoa irá gastar?

Modelaremos o problema no Solver do PuLP, portanto, possuímos:

```
import numpy as np
import pulp

# Definindo o problema como de minimizacao
prob = pulp.LpProblem('Problema da Dieta', pulp.LpMinimize)

# Definindo as variaveis de decisao
x1 = pulp.LpVariable('Pao Integral', lowBound=0, cat='Continuous')
x2 = pulp.LpVariable('Queijo Cottage', lowBound=0, cat='Continuous')
x3 = pulp.LpVariable('Mamao', lowBound=0, cat='Continuous')
x4 = pulp.LpVariable('Nozes', lowBound=0, cat='Continuous')
x5 = pulp.LpVariable('Salada Crua', lowBound=0, cat='Continuous')
x6 = pulp.LpVariable('Feijao', lowBound=0, cat='Continuous')
x7 = pulp.LpVariable('Arroz Integral', lowBound=0, cat='Continuous')
x8 = pulp.LpVariable('Frango Grelhado', lowBound=0, cat='Continuous')
x9 = pulp.LpVariable('Maca', lowBound=0, cat='Continuous')
x10 = pulp.LpVariable('Tapioca', lowBound=0, cat='Continuous')
x11 = pulp.LpVariable('Ovo', lowBound=0, cat='Continuous')
x12 = pulp.LpVariable('Atum Ralado', lowBound=0, cat='Continuous')
x13 = pulp.LpVariable('Iogurte', lowBound=0, cat='Continuous')

#Definindo a funcao a ser maximizada
MIN = 0.93*x1 + 5.24*x2 + 1.29*x3 + 3.05*x4 + 0.5*x5 + 0.63*x6 + 0.09*x7 + 0.89*x8 +
    1.95*x9 + 0.21*x10 + 0.99*x11 + 0.61*x12 + 1.69*x13

#Adicionando a funcao-objetivo
prob += MIN

#Definindo a funcao da Energia
energia = 136*x1 + 85*x2 + 45*x3 + 100*x4 + 25*x5 + 132*x6 + 25*x7 + 165*x8 + 72*x9 +
    68*x10 + 77*x11 + 17*x12 + 99*x13

#Adicionando a funcao da Energia nas restricoes
prob += (energia>=1870)

#Definindo a funcao da Proteina
proteina = 4.6*x1 + 17.2*x2 + 0.8*x3 + 2.3*x4 + 1.5*x5 + 8.8*x6 + 0.5*x7 + 31*x8 +
    0.3*x9 + 0*x10 + 6.2*x11 + 3*x12 + 3.9*x13

#Adicionando a funcao da Proteina nas restricoes
prob += (proteina>=46)

#Definindo a funcao do Calcio
calcio = 0.06*x1 + 0.03*x2 + 0.02*x3 + 0*x4 + 0.02*x5 + 0.03*x6 + 0*x7 + 0.02*x8 +
    0.01*x9 + 0*x10 + 0.03*x11 + 0*x12 + 0.14*x13

#Adicionando a funcao do Calcio nas restricoes
prob += (calcio>=1)

#Definindo a funcao do Sodio
sodio = 0.276*x1 +0.013*x2 + 0.0055*x3 + 0*x4 + 0.08*x5 + 0.001*x6 + 0*x7 + 0.074*x8 +
     0.001*x9 + 0.037*x10 + 0.139*x11 + 0.069*x12 + 0.053*x13

#Adicionando a funcao da Sodio nas restricoes
prob += (sodio>=1.5)

#Definindo a funcao da Ferro
ferro = 0.1*x1 + 0.01*x2 + 0.01*x3 + 0*x4 + 0.03*x5 + 0.12*x6 + 0*x7 + 0.06*x8 + 0.01*
```

```
    x9 + 0.02*x10 + 0.03*x11 + 0*x12 + 0*x13

#Adicionando a funcao da Ferro nas restricoes
prob += (ferro>=0.018)

#Definindo a funcao das Vitaminas
vitaminas = 0*x1 + 0.01*x2 + 0.77*x3 + 0*x4 + 0.93*x5 + 0*x6 + 0*x7 + 0*x8 + 0.12*x9 +
    0*x10 + 0.06*x11 + 0*x12 + 0.02*x13

#Adicionando a funcao das Vitaminas nas restricoes
prob += (vitaminas>=0.775)

#escrevendo o problema de otimizacao linear
print(prob)

# Resolvendo o problema
optimization_result = prob.solve()

# Verificando se a solucao otima foi encontrada
assert optimization_result == pulp.LpStatusOptimal

#mostrando o resultado
for var in (x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11, x12, x13):
    print('Obtencao da quantidade minima de {}: {:1.0f}'.format(var.name, var.value())
        )

OUTPUT:

Problema_da_Dieta:
MINIMIZE
0.09*Arroz_Integral + 0.61*Atum_Ralado + 0.63*Feijao + 0.89*Frango_Grelhado + 1.69*
    Iogurte + 1.95*Maca + 1.29*Mamao + 3.05*Nozes + 0.99*Ovo + 0.93*Pao_Integral +
    5.24*Queijo_Cottage + 0.5*Salada_Crua + 0.21*Tapioca + 0.0
SUBJECT TO
_C1: 25 Arroz_Integral + 17 Atum_Ralado + 132 Feijao + 165 Frango_Grelhado
 + 99 Iogurte + 72 Maca + 45 Mamao + 100 Nozes + 77 Ovo + 136 Pao_Integral
 + 85 Queijo_Cottage + 25 Salada_Crua + 68 Tapioca >= 1870

_C2: 0.5 Arroz_Integral + 3 Atum_Ralado + 8.8 Feijao + 31 Frango_Grelhado
 + 3.9 Iogurte + 0.3 Maca + 0.8 Mamao + 2.3 Nozes + 6.2 Ovo + 4.6 Pao_Integral
 + 17.2 Queijo_Cottage + 1.5 Salada_Crua >= 46

_C3: 0.03 Feijao + 0.02 Frango_Grelhado + 0.14 Iogurte + 0.01 Maca
 + 0.02 Mamao + 0.03 Ovo + 0.06 Pao_Integral + 0.03 Queijo_Cottage
 + 0.02 Salada_Crua >= 1

_C4: 0.069 Atum_Ralado + 0.001 Feijao + 0.074 Frango_Grelhado + 0.053 Iogurte
 + 0.001 Maca + 0.0055 Mamao + 0.139 Ovo + 0.276 Pao_Integral
 + 0.013 Queijo_Cottage + 0.08 Salada_Crua + 0.037 Tapioca >= 1.5

_C5: 0.12 Feijao + 0.06 Frango_Grelhado + 0.01 Maca + 0.01 Mamao + 0.03 Ovo
 + 0.1 Pao_Integral + 0.01 Queijo_Cottage + 0.03 Salada_Crua + 0.02 Tapioca
 >= 0.018

_C6: 0.02 Iogurte + 0.12 Maca + 0.77 Mamao + 0.06 Ovo + 0.01 Queijo_Cottage
 + 0.93 Salada_Crua >= 0.775

VARIABLES
Arroz_Integral Continuous
Atum_Ralado Continuous
Feijao Continuous
Frango_Grelhado Continuous
Iogurte Continuous
```

```
Maca Continuous
Mamao Continuous
Nozes Continuous
Ovo Continuous
Pao_Integral Continuous
Queijo_Cottage Continuous
Salada_Crua Continuous
Tapioca Continuous

Obtencao da quantidade minima de Pao_Integral: 12
Obtencao da quantidade minima de Queijo_Cottage: 0
Obtencao da quantidade minima de Mamao: 0
Obtencao da quantidade minima de Nozes: 0
Obtencao da quantidade minima de Salada_Crua: 1
Obtencao da quantidade minima de Feijao: 0
Obtencao da quantidade minima de Arroz_Integral: 0
Obtencao da quantidade minima de Frango_Grelhado: 0
Obtencao da quantidade minima de Maca: 0
Obtencao da quantidade minima de Tapioca: 0
Obtencao da quantidade minima de Ovo: 0
Obtencao da quantidade minima de Atum_Ralado: 0
Obtencao da quantidade minima de Iogurte: 2
```

Portanto, pelo Solver do PuLP, temos uma determinada resolução para um caso de problema da dieta, com uma minimização do custo de cada um dos produtos listados e de cada uma das restrições.

4 Resultados

Para cada base de dados dada para uma determinada dieta, há uma solução distinta, ou seja, seja uma tabela com n alimentos, possuindo m-1 restrições e uma lista de dados para a função objetivo que queremos minimizar, possuímos $n \times m$ dados. Na base de dados utilizada por [5], há 13 alimentos distintos, e por consequência, 13 variáveis de decisão, 7 restrições e uma lista de preços para ser utilizada na função objetivo, portanto há um total de 104 dados em uma dada matriz $M \in \mathbb{M}_{13\times 8}(\mathbb{R})$. Utilizando o método do solver PuLP do Python, obtemos, para a tabela dada, uma solução de x_1=12, x_2=0, x_3=0, x_4=0, x_5=1, x_6=0, x_7=0, x_8=0, x_9=0, x_{10}=0, x_{11}=0, x_{12}= e x_{13}=2, ou seja, a solução ótima foi alcancaçada (Comando pulp.LpStatusOptimal) e ela é dada por essas variáveis de decisão. O problema da dieta mesmo podendo ser modelado de outra forma (Por exemplo, colocar umas das restrições na função objetivo e pretender maximizar a quantidade da restrição com um determinado preço mínimo), entretanto ele não recusa o padrão linear das funções de restrição, mesmo com outra análise. Para o caso estudado neste projeto, há um certo padrão pois busca um balanço da vida saudável em detrimento do preço de cada alimento.

Como bem sabemos, o PuLP é um pacote do Python que se responsabiliza por resolver Problemas de Programação Linear (PPL) e ele utiliza o método do Simplex de iteração para encontrar os "vértices"de cada uma das restrições, iterar e encontrar o valor dito como ótimo dentro da função objetivo. E o método se deu como efetivo a partir do momento que resolveu sem dificuldade alguma o problema da Dieta dado pela base de dados adquirida. Portanto, em questão de viabilidade e facilidade para se utilizar um método para a solução de um PPL, o PuLP se mostrou efetivo dentro da proposta que ele prometia mostrar e também de mais fácil entendimento em relação ao método Simplex, mesmo que o programa tenha intrínseco dentro de suas iterações o mesmo, visto que o PuLP possui uma interface de melhor entendimento e envolve apenas a modelagem e a programação do problema, não a solução do mesmo.

5 Conclusão

Na primeira etapa, ao estabelecermos os conceitos de um Problema de Programação Linear (PPL), seguimos para a apresentação de métodos para sua resolução. Estudamos e descrevemos três métodos de suma importância neste assunto: o Método do Gráfico, que soluciona um PPL realizando uma análise dos pontos extremos do gráfico extraído deste PPL, o Método Simplex, que retorna a solução ótima através de iterações de um algoritmo e o Solver PuLP (Python), que, após a implementação do PPL em forma de algoritmo, utiliza o Método dos Pontos Interiores e o Método Simplex para a obtenção da solução ótima. Sendo este último método, o que utilizamos posteriormente para resolução do assunto central.

Em seguida, caracterizamos o Problema da Dieta, o qual é um dos pioneiros na programação linear. Atualmente esta temática ainda possui grande importância, visto que o consumo de alimentos processados aumentou, entretanto muitos desses alimentos são prejudiciais à saúde, o que serve de motivação para buscar uma alimentação saudável e barata. Situados e possuindo os conhecimentos adquiridos na primeira etapa, obtemos uma aplicação prática do Problema da Dieta, seguindo a tabela de [5]. Após isso, modelamos o problema em um PPL e o solucionamos utilizando o Solver do PuLP.

Por fim, apresentamos a solução ótima obtida do Problema da Dieta aplicado. Além disso, realizamos uma análise do PuLP, concluindo que este é um formidável método para obtenção da solução ótima de um PPL, pela sua facilidade de compreensão e execução, e pela efetividade do método.

Referências

[1] T. CRILLY. *50 ideias de matemática que você precisa conhecer.* São Paulo: Planeta do Brasil, 2017.

[2] Stuart Mitchell et al. Optimization with pulp. `https://coin-or.github.io/pulp/index.html`, 2003. Accessed: 2020-10-06.

[3] Marco Cesar Goldbarg and Henrique Pacca L Luna. *Otimização combinatória e programação linear: modelos e algoritmos.* Elsevier, 2005.

[4] Erico LISBOA. Apostila de pesquisa operacional, 2002, 2015.

[5] Giulia S. P. C. Tirone. Programação linear: uma aplicação ao problema da dieta. page 70, November 2019.