Princípio do Mínimo da Energia Potencial Total aplicado a deformação de uma estrutura treliçada do tipo cantilever

Luis Vinicius Costa Silva

26 de Junho de 2019

Disciplina: Otimização Clássica
Professor: Romes Antônio Borges

1 Introdução

Este trabalho tem como objetivo criar um modelo matemático que permita representar a deformação de uma estrutura dada sua configuração inicial. Foi utilizada uma abordagem da otimização clássica que se baseia no Princípio do mínimo da Energia Potencial.
Para tanto, fez-se a modelagem de uma viga treliçada do tipo cantilever, e a obtenção das equações constitutivas do sistema e a escrita do problema em forma de problema de otimização. Posteriormente foi escrito um código computacional (em linguagem Python) para a resolução computacional do problema, isto é: dada uma configuração inicial da estrutura, minimizar a Função de Energia Potencial Total do sistema, obtendo os valores das variáveis de estados (configuração final da estrutura) associada a energia potencial mínima do sistema.
Foram utilizados três algoritmos clássicos para a resolução do problema de otimização: L-BFGS-B, SLSQP e Newton Truncado. Além disso foi utilizado a Evolução Diferecial como algoritmo evolutivo para a resolução do problema. Os resultados obtidos por tais algoritmos foram comparados em um estudo de caso, avaliando as vantagens e desvantagens de cada um deles para o problema em questão.

2 Modelagem

2.1 Princípio do mínimo da Energia Potencial Total

O Princípio do mínimo da Energia Potencial dita que o estado de equilíbrio mecânico de um sistema é aquele que (de todas as alternativas possíveis) tem a menor energia potencial (basicamente uma reformulação da segunda lei da

Termodinâmica). A determinação da configuração final de um sistema mecânico implica na formulação da função de energia potencial total do sistema em função das forças internas (energia armazenada por um sistema que sofre deformação - strain) e a energia potencial das forças externas. É sabido que a Energia Potencial Total de um sistema é dado pela diferença entre as forças internas (strain Energy/deformação) e o trabalho das Forças Externas.

$$\pi = U - W \tag{1}$$

Expandindo os termos da equação anterior, temos que:

$$\pi = \sum_{e=1}^{n} \Lambda^{(e)} - \sum_{i=1}^{m} F_i u_i \tag{2}$$

Onde $\Lambda^{(e)}$ é a energia de deformação para cada elemento do sistema e $F_i u_i$ é a força externa aplicada em cada campo de deslocamento do sistema.
As forças internas/deformação em cada elemento é dada por:

$$\Lambda^{(e)} = \int_V \frac{1}{2} \sigma \epsilon \, dV \tag{3}$$

Aplicando a Lei de Hook, temos:

$$\Lambda^{(e)} = \int_V \frac{1}{2} E \epsilon^2 \, dV \tag{4}$$

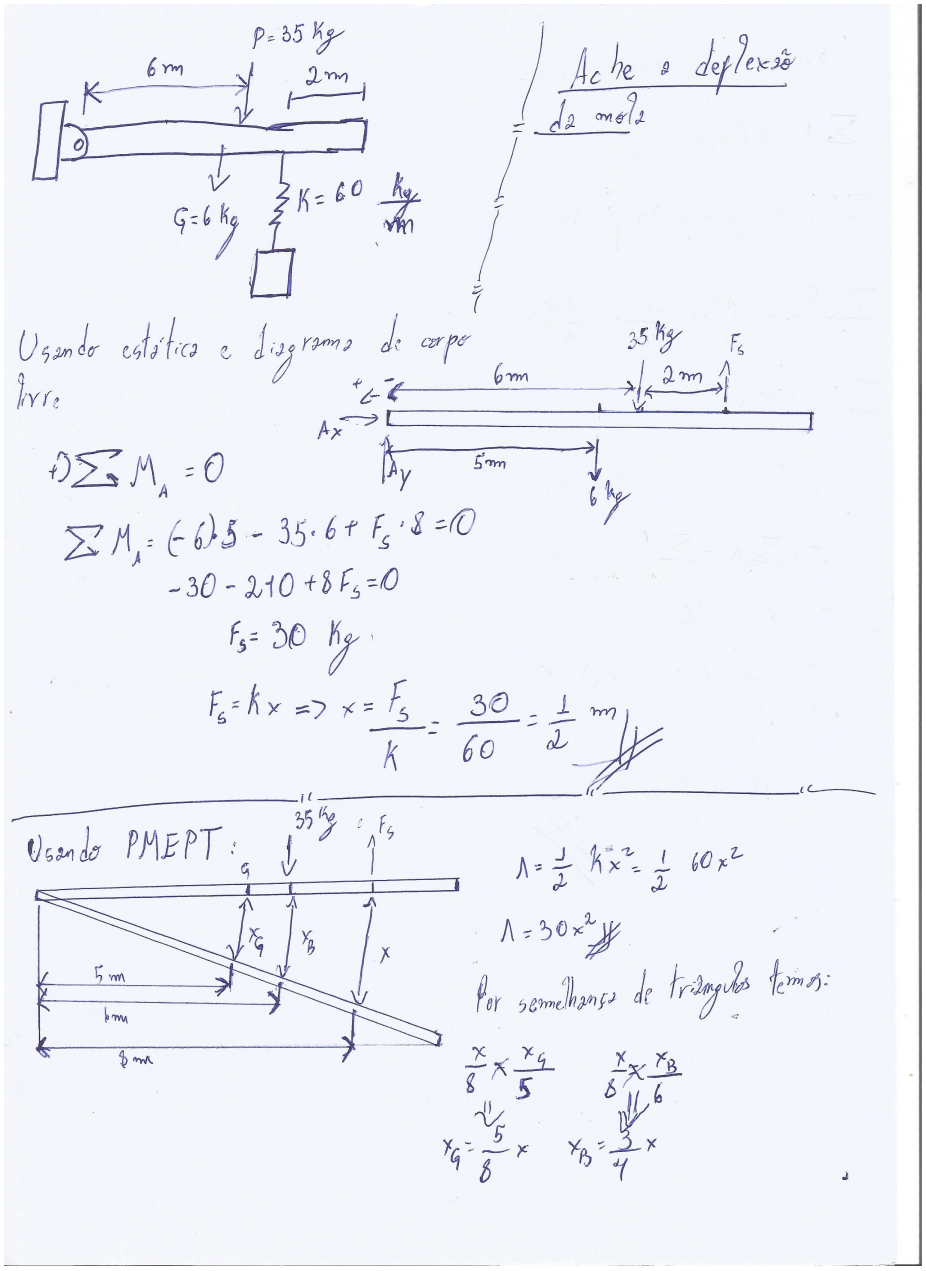
Ou seja, as forças internas são dadas pela área delimitada pela curva $\sigma - \epsilon$ (curva tensão-deformação) multiplicado pelo volume do material.
A energia potencial mínima do sistema pode ser obtida igualando a derivada da energia potencial pelo deslocamento a zero:

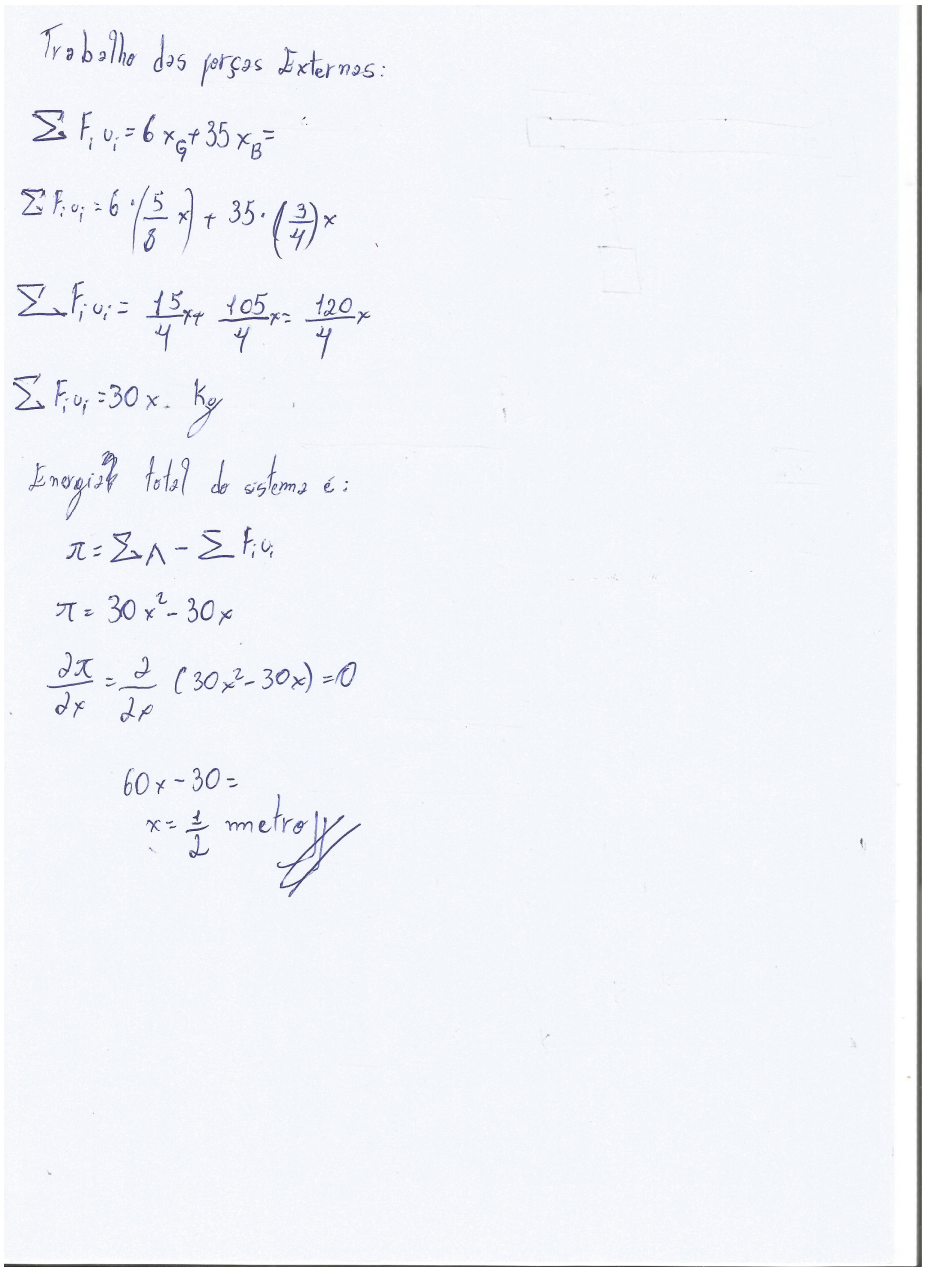
$$\frac{\partial}{\partial u_i} \sum_{e=1}^{n} \Lambda^{(e)} - \frac{\partial}{\partial u_i} \sum_{i=1}^{m} F_i u_i = 0, \;\; i = 1, 2, ..., n \tag{5}$$

O princípio do mínimo da Energia Potencial Total se faz útil pelo fato de ser computacionalmente mais rápido que uma integração de equação diferencial, visto que são necessários computar estados intermediários do sistema até atingir o estado de equilíbrio, acarretando em mais uso de processamento e memória. Além disso, a resolução deste tipo de problema por este princípio requer apenas a primeira derivada, além de incorporar as condição de contorno de força (condição de contorno natural,i.e: forças, momentos prescritos, etc.) automaticamente. O deslocamento admissível precisa satisfazer somente a condição de contorno de deslocamento (condição de contorno geométrica).

Visto que tal diferenciação torna-se muito complicada para sistemas com vários u_i (i.e: pontos de deslocamento), faz-se necessário a formulação do problema no formato de minimização, a fim de que algoritmos de otimização atinjam o resultado esperado. A próxima seção demonstra a aplicação do Príncipio do mínimo da Energia Potencial Total para um sistema massa-mola simples.

2.2 Exemplo analítico do Príncipio do mínimo da Energia Potencial Total





2.3 Modelagem da treliça simplesmente engastada

O modelo busca representar a deformação de uma estrutura engastada em apenas um dos lados (cantilever) utilizando-se de uma abordagem de otimização para este fim. Logo, é necessário escrever uma função objetivo que relacione as

variáveis de estado do sistema a energia potencial do sistema. Considera-se que a estrutura é um sistema massa-mola discretizado em forma de malha, com as seguintes características:

- O sistema possui 2 graus de liberdade;
- Considera-se que a viga é constituída de n massas (nós), conectadas por $2n-3$ molas (elementos de malha triangulares);
- Rigidez da estrutura e módulo de elasticidade são definidos em função da rigidez e comprimento relaxado das molas.
- As molas podem se deformar infinitamente (i.e: o modelo não representa rompimento).

Logo, a energia potencial total do sistema é dada pela soma da energia potencial elástica e a energia potencial gravitacional de cada/entre cada nó. Abaixo foram listadas as 2 equações fundamentais do modelo.

Energia potencial de uma mola entre dois pontos:

$$\frac{1}{2}k((|p_1 - p_2|) - r)^2 \tag{6}$$

Energia potencial total de uma massa:

$$E_{pg} = mgh \tag{7}$$

Para um sistema com n massas, generaliza-se os índices das fórmulas, e, obtêm-se a seguinte função objetivo:

$$\begin{aligned} min \;\; U = \sum_{i=1}^{n} mgy_i + \frac{1}{2}\sum_{i,j=1}^{n} ka_{ij}(|p_i - p_j| - l_{ij})^2 \;\; s.t: \\ p_1 = (0.0, 0.0) \\ p_2 = (0.0, -0.1) \\ p_i - 1.0 \leq p_i \leq p_i + 1.0 \;, i = 3, ..., n, \;\; (opcional) \end{aligned} \tag{8}$$

- n – quantidade de nós da malha;
- m – massa sobre a estrutura (kg);
- g – constante gravitacional (m/s);
- y_i – coordenada y do nó (matriz coluna);
- k – constante elástica das molas (kg/m);
- p_i – i-ésima posição (x,y) do nó (matriz);
- a_{ij} – elemento da matriz de adjacência (1 caso exista mola, 0 senão);

- l_{ij} – comprimento relaxado da mola (i,j) (0 caso não exista mola entre nós) (matriz);

As restrições do problema de otimização representam as condições de contorno do sistema.

3 Algoritmos utilizados

3.1 L-BFGS-B

O algoritmo L-BFGS-B (Byrd et al., 1995) é uma modificação do algoritmo BFGS capaz de lidar com limites laterais (i.e: $a \leq x \leq b$) diretamente. Além disso, este algoritmo não armazena uma aproximação densa da inversa da matriz hessiana, ao invés disso, o algoritmo armazena apenas alguns vetores que representam tal termo implicitamente, i.e: o algoritmo armazena as últimas m atualizações de x e $\nabla f(x)$. Estes termos são utilizados quando algumas operações exigem o produto vetorial da matriz H_k.
O tratamento dos limites laterais é realizado através do método de gradiente que identifica o conjunto com os limites das variáveis de entrada que estão ativas, e então o modelo quadrático é aproximadamente minimizado, em relação as variáveis livres.
Após isso, o próximo ponto de avaliação é calculado através da busca unidimensional, utilizando um determinado tamanho de passo $\alpha_{l-bfgs-b}$ (hiperparâmetro).

3.2 SLSQP

O SLSQP (Sequential Least Squares Quadratic Programming) é um algoritmo de programação sequencial de mínimos quadrados. O SLSQP usa o algoritmo BFGS para aproximar a Hessiana e a função de mérito L é usada para uma busca unidimensional.
As restrições de desigualdade e igualdade são combinadas e expressadas por $\overrightarrow{h_k}$, o vetor de multiplicadores de Lagrange é dado por u_k. As restrições de desigualdade são igualadas a zero e tratadas de acordo (entretanto os multiplicadores de Lagrange associados a estas restrições não devem ser negativos), restrições de desigualdade inativas são ignoradas.
Existem 3 critérios de parada para o algoritmo SLSQP, são eles:

$$|\nabla f_k^T d_k| + \sum |u_i||h_i(x_k)| \leq tol \tag{9}$$

Equation 9: Condição de Parada 1

$$|f(x)_k - f(x)_{k+1}| \leq tol \tag{10}$$

Equation 10: Condição de Parada 2

$$k \geq \max \text{ it} \tag{11}$$

Equation 11: Condição de Parada 3

Como pode ser visto pela equação **??** O tamanho do primeiro passo da busca unidimensional geralmente é 1, este tamanho de passo é reduzido sucessivamente por um determinado fator de contração. Logo, o primeiro termo $|\nabla f_k^T d_k|$ representa representa o quanto de melhoria a função objetivo obteve ao longo da direção de descida mais íngreme com o maior comprimento de passo. O segundo termo representa uma combinação ponderada do total de todas as violações. Portanto, este critério denota que, uma vez que ocorreu uma melhoria significativa da função Lagrangeana L menor que uma tolerância pré-definida, então considera-se que o algoritmo convergiu para um $x*$.

O critério de parada **??** e **??** são amplamente conhecidos, e denotam respectivamente a parada do algoritmo em função da diferença entre dois x_k de iterações consecutivas ser menor que uma tolerância pré-definida, e a parada do algoritmo após um número máximo de iterações.

A função de mérito L_1 (equação 12) é utilizada para garantir uma convergência global, o que significa que o x ótimo será encontrado, independentemente do x inicial. A função de mérito pode ser considerada como uma função objetivo para a busca unidimensional, esta é decrementada sucessivamente para a geração de um novo x_k no decorrer do processo. Para problemas irrestritos a função objetivo original pode ser usada como função de mérito, em qualquer outro caso, a função de mérito L_1 assume a seguinte forma:

$$L_1 = f(x_k + \alpha d_k) + \sum_{j=1}^{m} \rho_j |h_i(x_k + \alpha d_k)| \tag{12}$$

Equation 12: Função de mérito L1

O fluxograma abaixo demonstra o funcionamento básico do algoritmo SLSQP:

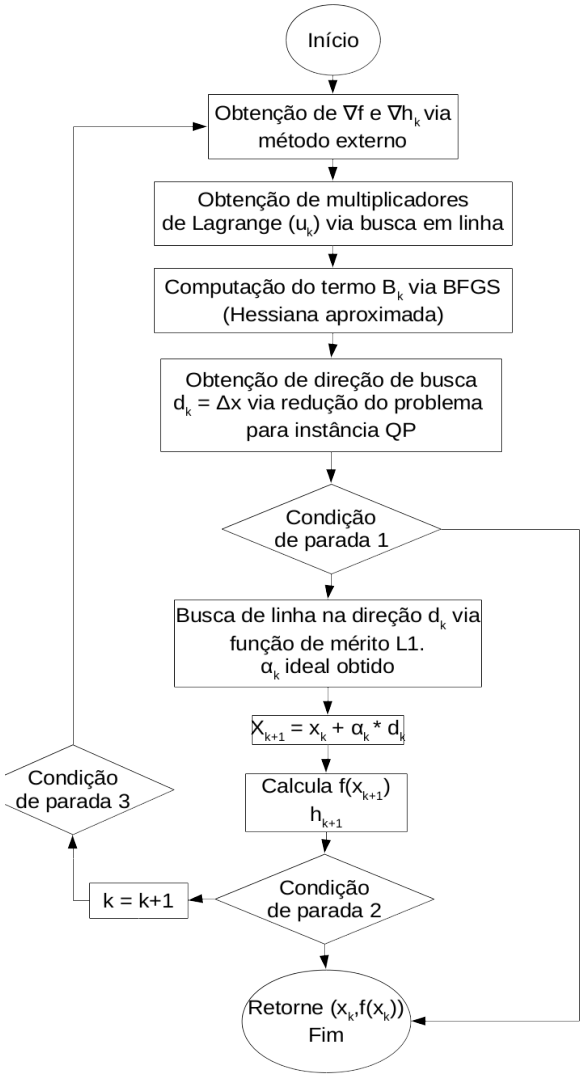


Figura 1: Fluxograma do algoritmo SLSQP

3.3 Método de Newton Truncado

O método de Newton truncado é um algoritmo de otimização que, dado um x_k, aproxima um $x*$ resolvendo as equações de Newton (...) usando algum algoritmo iterativo com precondicionamento M^k. M^k é escolhido com informação de algum outro algoritmo de otimização (geralmente gradiente conjugado) ou com base em iterações anteriores. Usando a solução obtida para as equações de Newton, uma direção de descida é computada, e uma nova aproximação x_{k+1} de $x*$ é obtida. As equações de Newton são definidas da seguinte forma:

Considerando que a função objetivo é aproximada por uma série de Taylor, temos:

$$F(x^{(0)} + p) = F(x^{(0)}) + p^T \nabla F(x^{(0)} + \frac{1}{2} p^T \nabla^2 F(x^{(0)}) p + R_3(x^{(0)}, p) \qquad (13)$$

onde $R_3(x^{(0)}, p)$ representam os termos de alta ordem da série. Logo, as equações de Newton são dadas pela fórmula abaixo:

$$Hp = -g \qquad (14)$$

Onde H é a matriz Hessiana, g é o gradiente de f e p é o ponto de interesse da série.

3.4 Evolução Diferencial

O algoritmo de Evolução Diferencial é um algoritmo de otimização simples e eficiente que foi proposto por Rainer Storn e Kenneth Price em 1995 (Storn e Price, 1995). Mostra-se eficaz para funções objetivo que não são diferenciáveis ou convexas e tem facilidade na busca do ótimo com populações pequenas (Cheng e Hwang, 2001). O DE pode ser descrito como uma manipulação de indivíduos que representam as soluções candidatas. No decorrer das gerações, essas soluções candidatas sofrem modificações de mutação e cruzamento, onde são geradas novas soluções candidatas, e logo após é feita a seleção e o ciclo se repete. A figura 2 demonstra o funcionamento básico do algoritmo de Evolução Diferencial

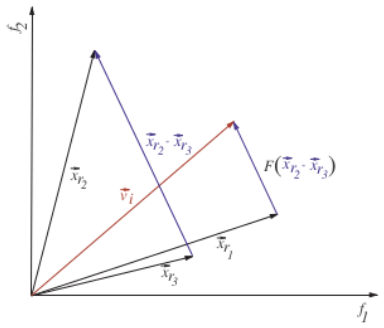


Figura 2: Funcionamento da Evolução Diferencial – Price et al. (2005)

Partindo de três vetores $\overrightarrow{x_{r1}}$, $\overrightarrow{x_{r2}}$ e $\overrightarrow{x_{r3}}$, são escolhidos aleatoriamente dois deles (nesse caso $\overrightarrow{x_{r2}}$ e $\overrightarrow{x_{r3}}$), sendo realizada a subtração dos mesmos. O resultado é multiplicado por um escalar F , gerando assim um vetor com módulo diferente da subtração original. Esse novo vetor é então somado ao vetor $\overrightarrow{x_{r1}}$, fornecendo assim um novo vetor $\overrightarrow{v_i}$. Esse vetor $\overrightarrow{v_i}$ indicará uma nova posição no espaço, ou um novo indivíduo (muito semelhante a um algoritmo genético, mas com cromossomos compostos por valores reais).

4 Resultados

Para a documentação neste relatório, foi elaborado o experimento abaixo:

Parâmetros do experimento 1:

- massa – $m = 0.1$;
- número de nós – $n = 20$;
- distância inicial entre nós – $e = 0.1$;
- comprimento das molas em situações sem deformação – $l = 0.1$;
- constante de rigidez da mola $k = 10000$

As próximas seções documentam os resultados obtidos.

4.1 Métodos Clássicos

Configuração inicial da treliça:

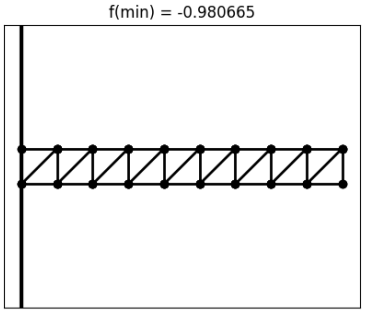


Figura 3: Representação gráfica da configuração inicial da treliça

Métodos x Métricas	Qtde de variáveis	Qtde de Iterações	Avaliações da função objetivo	Min F(x)
L-BFGS-B	40 (20 nós)	133	138	-2.52938265048
SLSQP	40 (20 nós)	58	2557	-2.529382692846
Newton Truncado	40 (20 nós)	830	62	-2.51249006635

Tabela 1: Resultados obtidos no experimento 1

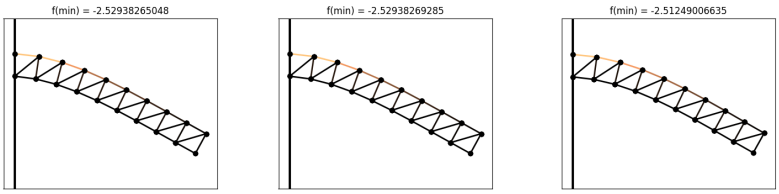


Figura 4: Configuração Final da treliça computada pelo Algoritmo L-BFGS-B, SLSQP e Newton Truncado respectivamente

x	y
0.00000000000000000e+00	0.00000000000000000e+00
0.00000000000000000e+00	-1.00000000000000056e-01
1.07660496345496181e-01	-1.23480881396893591e-02
9.28560934442665686e-02	-1.12132631249419398e-01
2.11045383683019311e-01	-3.84644753155233207e-02
1.85055803111605188e-01	-1.36025518741883456e-01
3.10061083866447540e-01	-7.38598911778323951e-02
2.75876147563759278e-01	-1.68840871542965842e-01
4.05225892142389021e-01	-1.15495348155972274e-01
3.65274699789626622e-01	-2.08106203574760312e-01
4.97352261909840776e-01	-1.61086575594408504e-01
4.53549333043553271e-01	-2.51800545754000127e-01
5.87305406194069013e-01	-2.08938358868559909e-01
5.41124211270353039e-01	-2.98298006330763865e-01
6.75876188311249959e-01	-2.57816983643167352e-01
6.28412700750138886e-01	-3.46318950372309364e-01
7.63717784643658693e-01	-3.06863136706818251e-01
7.15734767607390743e-01	-3.94892541262722219e-01
8.51319622158413170e-01	-3.55542215889908198e-01
8.03269509346520083e-01	-4.43339493748393043e-01

Tabela 2: Posições finais dos nós computados pelo L-BFGS-B

x	y
0.00000000000000000e+00	0.00000000000000000e+00
0.00000000000000000e+00	-1.00000000000000056e-01
1.07658101658850966e-01	-1.23425287718744482e-02
9.28569790563922443e-02	-1.12127372321176325e-01
2.11043548390084962e-01	-3.84526116916297719e-02
1.85058960761004226e-01	-1.36013970930589728e-01
3.10061495086762673e-01	-7.38375963920043510e-02

Continuação da tabela 3

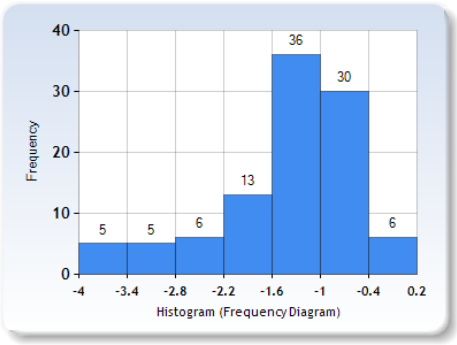
x	y
2.758808407504163562e-01	-1.688194012225558427e-01
4.052274160172885109e-01	-1.154721206354224761e-01
3.652794197173069302e-01	-2.080853121149189855e-01
4.973554329820704112e-01	-1.610579591497134644e-01
4.535571696553672427e-01	-2.517734303334946966e-01
5.873107838861030272e-01	-2.089042278612215919e-01
5.411329922867171849e-01	-2.982663049064730454e-01
6.758827785101579888e-01	-2.577831678481600219e-01
6.284233764303035485e-01	-3.462874535685726007e-01
7.637250577912088056e-01	-3.068229353638805001e-01
7.157473943252998794e-01	-3.948549020339718885e-01
8.513293735802619500e-01	-3.554980033699064457e-01
8.032843643335488837e-01	-4.432975511277799652e-01

Tabela 3: Posições finais dos nós computados pelo SLSQP

x	y
0.000000000000000000e+00	0.000000000000000000e+00
0.000000000000000000e+00	-1.000000000000000056e-01
1.073453320603787731e-01	-1.131906141904176315e-02
9.390695378617000544e-02	-1.111144547211459221e-01
2.109855703932395077e-01	-3.505453679425795527e-02
1.871162548763355415e-01	-1.329081016022125861e-01
3.109290112846055387e-01	-6.796005448866306031e-02
2.786578919106891261e-01	-1.634131799516077566e-01
4.072288059110498737e-01	-1.080791236868817312e-01
3.685794437575616311e-01	-2.012766910145951105e-01
5.002811533915770825e-01	-1.532024195396144706e-01
4.570264280056981487e-01	-2.440769482580765759e-01
5.908554038684504794e-01	-2.009954397605685228e-01
5.443707874541260372e-01	-2.902751946570642505e-01
6.792081549818771435e-01	-2.504898803189012924e-01
6.309492576946710285e-01	-3.387666102692873382e-01
7.663160328145873779e-01	-3.007088386085366438e-01
7.175186124147107103e-01	-3.884533589826867916e-01
8.533262796408570550e-01	-3.504586983865930483e-01
8.045345136075724435e-01	-4.377923103295616736e-01

Tabela 4: Posições finais dos nós computados pelo Método de Newton Truncado

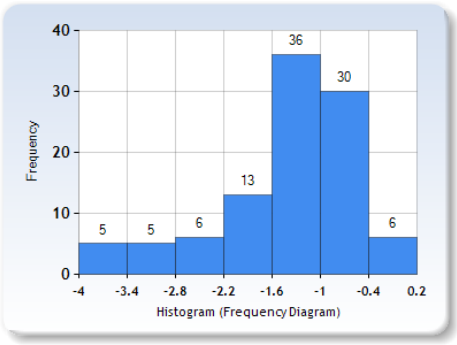
4.2 Métodos Evolutivos



Frequency Table	
Class	*Count*
-4--3.400001	5
-3.4--2.800001	5
-2.8--2.200001	6
-2.2--1.600001	13
-1.6--1.000001	36
-1--0.400001	30
-0.4-0.199999	6

Your Histogram	
Mean	-1.43625
Standard Deviation (s)	0.83188
Skewness	-0.97257
Kurtosis	0.64226
Lowest Score	-3.731048
Highest Score	-0.11218
Distribution Range	3.61887
Total Number of Scores	101
Number of Distinct Scores	101
Lowest Class Value	-4
Highest Class Value	0.199999
Number of Classes	7
Class Range	0.6

Figura 5: Histograma e análise estatística da resposta do algoritmo de Evolução Diferencial para uma população de 40 candidatos



Frequency Table	
Class	*Count*
-4--3.400001	5
-3.4--2.800001	5
-2.8--2.200001	6
-2.2--1.600001	13
-1.6--1.000001	36
-1--0.400001	30
-0.4-0.199999	6

Your Histogram	
Mean	-1.43625
Standard Deviation (s)	0.83188
Skewness	-0.97257
Kurtosis	0.64226
Lowest Score	-3.731048
Highest Score	-0.11218
Distribution Range	3.61887
Total Number of Scores	101
Number of Distinct Scores	101
Lowest Class Value	-4
Highest Class Value	0.199999
Number of Classes	7
Class Range	0.6

Figura 6: Histograma e análise estatística da resposta do algoritmo de Evolução Diferencial para uma população de 100 candidatos

4.3 Discussões

Dentre os algoritmos clássicos, nota-se que o algoritmo L-BFGS-B obteve uma performance superior ao Método de Newton Truncado, visto que este obteve o melhor trade-off entre número de iterações e número de avaliações da função objetivo. Ao passo que o L-BFGS-B era superado pelo Newton Truncado em número de avaliações da função objetivo (45% a mais), este o superava em iterações (83% a menos). Similarmente o L-BFGS-B superou o SLSQP, visto que o L-BFGS-B levou 133 iterações para convergir ao ótimo, enquanto o SLSQP necessitou de apenas 58. Em contra-partida, o SLSQP necessitou de 2557 avaliações da função objetivo, enquanto que o L-BFGS-B necessitou de apenas 138 avaliações da função objetivo.
A superioridade do L-BFGS-B é mantida para configurações iniciais mais complexas, tornando-o o método ideal para a resolução deste tipo de problema.
No que tange a métodos evolutivos, a Evolução Diferencial foi escolhida devido ao fato deste algoritmo necessitar de pequenas populações (quando comparados a outros métodos evolutivos) para atingir a convergência. Em testes preliminares, outros algoritmos evolutivos necessitaram de muito tempo para gerar uma população de indíviduo (geralmente um vetor com 20 posições), e realizar ope-

rações de mutação e similares. Infelizmente o algoritmo de Evolução Diferencial não foi capaz de resolver o problema original, sendo necessário adicionar mais restrições laterais ao mesmo, da forma:

$$-i \mod 2 \leq p_{i+4} \leq (i \mod 2) - 1,\ i = 0, 1, ..., n-2 \tag{15}$$

Onde tais limites são relativos as posições iniciais de cada nó. Após isto, foram realizados dois testes com tamanhos de população de 40 e 100 candidatos. Em ambos os casos foi utilizada amostragem por hipercubo latino, a fim de que a população de vetores iniciais cobri-se o espaço de busca ao máximo. As taxas de recombinação foram respectivamente 0.7 e 0.5
Nota-se que o aumento da população e a diminuição da taxa de recombinação levou o algoritmo de Evolução Diferencial a convergir para um resultado correto com mais frequência, mesmo assim ambos os testes as respostas apresentaram uma dispersão substancial das respostas dadas pelo algoritmo evolutivo.

5 Conclusão

Concluí-se que o modelo desenvolvido é capaz de associar os estados das variáveis do sistema (campos de deslocamentos) a energiqa potencial total do sistema, possibilitando que o problema de computar a configuração final da estrutura seja feito na forma de problema de otimização. Nota-se que o algoritmo L-BFGS-B mostrou-se mais adequado para a resolução deste problema, visto que este foi capaz de realizar superar o algoritmo SQP no número de avaliações da função objetivo (sendo ligeiramente superado no número de iterações) assim como superar o método de Newton Truncado na quantidade de iterações necessárias para atingir a convergência (sendo ligeiramente superado na quantidade de avaliações da função objetivo). Todos os algoritmos clássicos foram capazes de computar a configuração final da estrutura sob diferentes configurações iniciais com uma margem de erro aceitável.

A Evolução Diferencial foi escolhida principalmente pelo fato de ser capaz de convergir ao x ótimo com populações relativamente pequenas (em comparação a outros métodos evolutivos), este fato foi claramente percebido nos testes, visto que cada solução candidata era um vetor de 40 variáveis, o que tornava lento o processo de geração de uma nova população. Dentre os algoritmos evolutivos previamente testados, apenas o algoritmo de Evolução Diferencial foi capaz de realizar a convergência para uma solução (não necessariamente correta) em tempo hábil para o projeto.

Mesmo assim, o algoritmo de Evolução Diferencial teve um resultado insatisfatório neste estudo, devido a necessidade de inserir restrições laterais adicionais a fim de assegurar uma convergência para um resultado correto. Como observado nos histogramas da seção de resultados, os resultados para o algoritmo de Evolução Diferencial foram substancialmente dispersos, até mesmo com o acréscimo

do tamanho da população e diminuição da taxa de recombinação/crossover. Além disso, seu custo computacional é alto, devido a grande quantidade de avaliações da função objetivo e geração de candidatos. Conjectura-se que uma mudança na estratégia de mutação do algoritmo (foi utilizada a *best1bin*) possa apresentar melhores resultados.