Submetido para TEMA

Projeto Ótimo de Treliças: Otimização Estrutural

FRANCISCO A. M. GOMES[1], TADAO ANDO Jr.[2] Departamento de Matemática Aplicada, IMECC, UNICAMP, 13081-970, Campinas, SP, Brasil.

Resumo. Treliças são estruturas compostas por barras delgadas e ligadas entre si por meio de rótulas, geralmente sujeitas somente a cargas nos nós, apresentando, neste caso, apenas esforços axiais. Neste trabalho, determinamos a geometria ótima de uma treliça sujeita a um conjunto de forças, obtendo a estrutura mais rígida possível que obedece a restrições de material e de domínio. A abordagem escolhida engloba os problemas de dimensionamento geométrico e topológico, resultando num modelo de programação não linear. Resolvemos esse problema adaptando um algoritmo do tipo restauração inexata, para o qual propomos especializações levando em conta as particularidades do modelo e o objetivo de resolver problemas de grande porte.

1. Introdução

O problema de otimização topológica de treliças consiste em encontrar a estrutura mais rígida que suporta um conjunto de forças, obedecendo a restrições de material e de domínio. Como descrito em [1], diferentes aspectos podem ser considerados ao projetar uma estrutura. Esses aspectos, em conjunto com características físicas desejadas na estrutura final (e.g. a minimização de uma grandeza física e quantidades limitadas de material), dão origem a problemas de otimização, dentre os quais citamos os de dimensionamento, forma e topologia.

Na sua forma mais básica, o problema de projeto estrutural pede que determinemos se cada ponto do espaço contem ou não material. Uma maneira de responder essa pergunta é discretizar o domínio em questão e analisar cada elemento da malha. Podemos então enxergar as treliças como uma discretização do problema de projeto estrutural geral. Neste trabalho, propomos uma especialização de um algoritmo do tipo restauração inexata para a determinação da topologia ótima de treliças.

Este texto foi dividido da seguinte forma: nas seções 2 e 3 descrevemos a abordagem escolhida para modelar o problema e a formulação adotada; o algoritmo do tipo restauração inexata utilizado é descrito na seção 4 e sua especialização para a resolução do problema de treliças é apresentada na seção 5; resultados numéricos preliminares que ilustram o bom comportamento do método proposto são mostrados na seção 6; finalmente, na seção 7 apresentamos uma conclusão, bem como diretrizes para trabalhos futuros.

[1] chico@ime.unicamp.br
[2] tadao@ime.unicamp.br

2. Estrutura Base

Para modelar o problema, usamos a abordagem conhecida como estrutura base (veja exemplo na Figura 1), na qual enxergamos um determinado subconjunto de conexões entre um conjunto fixo de nós como elementos em potencial da estrutura final, o que nos fornece um problema de otimização discreta. Neste cenário, treliças são convenientes pois permitem contornar as dificuldades do problema de programação inteira inerente a essa abordagem considerando como variáveis de projeto os volumes das barras (variáveis contínuas) e utilizando um volume nulo para representar a ausência de uma barra. Resolvemos assim, simultaneamente, os problemas de dimensionamento e topologia.

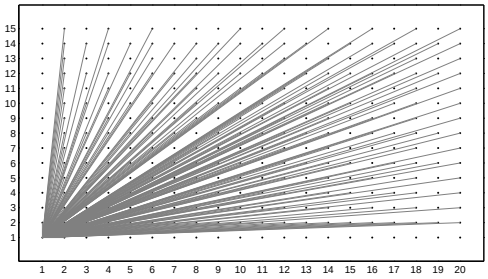

Figura 1: Exemplo de estrutura base para uma malha de 20 por 15 nós na qual todos estão conectados entre si por uma barra (nível de conexão máximo). Barras superpostas não são consideradas. Para manter a clareza da figura somente as barras que partem do nó inferior esquerdo foram representadas.

Neste trabalho consideramos domínios retangulares e definimos o universo de barras da estrutura base escolhendo um nível de conexão desejado para a estrutura. Não há restrições para a posição inicial dos nós. Entretanto, a forma do domínio não entra no problema de otimização. Deste modo, é conveniente utilizar um grande número de nós para reduzir o efeito negativo de desconsiderar essa flexibilidade adicional. Com isso, somos levados a considerar opções para resolver problemas de grande porte, entre elas a escolha modelos mais simples, reduzindo o número de variáveis, e a especialização do algoritmo.

3. Formulação do problema

Nos problemas descritos a seguir consideramos estruturas base com n nós, N graus de liberdade e m barras. Podemos ter dois ou três graus de liberdade por nó, se considerarmos estruturas planas ou espaciais, respectivamente. É necessário ainda restringir o movimento de alguns nós para prover apoios para a estrutura. Para uma treliça bidimensional temos $N < 2n$ e, tipicamente, $m = O(n^2)$. Os dados do problema são a carga de forças $f \in \mathbb{R}^N$, o volume total de material disponível

$V \in \mathbb{R}$ ($V > 0$) e a estrutura base. Usamos como variáveis de estado (que descrevem a situação da estrutura sob a ação das cargas) os deslocamentos nodais, $u \in \mathbb{R}^N$, e como variáveis de projeto os volumes da barras, representados por $t \in \mathbb{R}^m$. Finalmente, a cada barra corresponde ainda uma matriz de rigidez K_i que aparece nas restrições de equilíbrio estático da estrutura.

Como o trabalho é uma grandeza diretamente proporcional ao deslocamento que uma força causa em um corpo, podemos concluir que quanto mais rígida for uma estrutura, menor o trabalho que uma carga é capaz de realizar sobre ela. O termo $f^T u$ é usado para medir o trabalho total realizado por uma força externa f sobre uma treliça. Formulamos, então, o problema de determinar a treliça mais rígida para suportar uma carga f, com um volume de material V e com barras restritas a um subconjunto das barras que compõe a estrutura base, como

$$\begin{aligned} \min_{u\in\mathbb{R}^N,\ t\in\mathbb{R}^m} \quad & f^T u \\ \text{sujeito a} \quad & \left(\sum_{i=1}^{m} t_i K_i\right) u = f \\ & \sum_{i=1}^{m} t_i = V \\ & t_i \geq 0,\ i = 1 \dots m \end{aligned} \tag{3.1}$$

Demonstra-se em [2] que o problema (3.1) é equivalente à formulação na energia potencial

$$\begin{aligned} \max_{t\in\mathbb{R}^m} \min_{u\in\mathbb{R}^N} \quad & \left\{ \frac{1}{2}\ u^T \left(\sum_{i=1}^{m} t_i K_i\right) u - f^T u \right\} \\ \text{sujeito a} \quad & \sum_{i=1}^{m} t_i = V \\ & t_i \geq 0,\ i = 1 \dots m \end{aligned} \tag{3.2}$$

Seguindo o caminho proposto em [1], trocamos a posição dos operadores min e max no problema (3.2) e introduzimos uma nova variável para chegar à seguinte formulação (suave e convexa) somente nos deslocamentos nodais:

$$\begin{aligned} \min_{u\in\mathbb{R}^N,\ \alpha\in\mathbb{R}} \quad & \alpha - f^T u \\ \text{sujeito a} \quad & \frac{V}{2}\left[u^T K_i u\right] \leq \alpha,\ i = 1, \dots, m \end{aligned} \tag{3.3}$$

Para o caso de um único carregamento, o problema acima ainda pode ser reformulado como um problema de programação linear, como descrito em [1]. Temos por objetivo, entretanto, abordar o caso de múltiplos carregamentos, para o qual tal tipo de reformulação não é válida.

Vamos supor, então, que sejam dados M carregamentos $f^p,\ p = 1, \ldots, M$. Separando os vetores de deslocamento para cada uma das cargas e compondo, através de uma média ponderada, uma medida do trabalho total realizado pelas forças externas, podemos reescrever a formulação (3.3) como

$$\begin{aligned} \min_{u_p \in \mathbb{R}^N \ \alpha \in \mathbb{R}} \quad & \alpha - \sum_{p=1}^{M} w_p f_p^T u_p \\ \text{sujeito a} \quad & \frac{V}{2} \sum_{p=1}^{M} \left[w_p u_p^T K_i u_p \right] \leq \alpha,\ i = 1, \ldots, m \\ & \sum_{p=1}^{M} w_p = 1 \end{aligned} \tag{3.4}$$

onde $w_p,\ p = 1, \ldots, M$ são pesos atribuídos aos diversos carregamentos.

4. Restauração Inexata

No modelo descrito pelo problema (3.1), as restrições foram obtidas considerando as leis físicas que devem ser satisfeitas, salvo simplificações, para que uma treliça obtida como solução seja capaz de manter-se estável sob ação de uma carga fornecida. Na função objetivo associamos um custo, representado aqui pelo trabalho realizado pelas forças externas, a cada estado da estrutura. Sendo assim, fica evidente que o problema beneficia-se de métodos que privilegiem satisfazer as restrições em detrimento da obtenção do melhor valor do objetivo, pois uma estrutura que satisfaça as restrições de equilíbrio mas não seja a mais rígida ainda pode ser construída e desempenhar a função de transferir a carga para os apoios.

O método de Restauração Inexata descrito em [3], e referido de agora em diante como algoritmo IR, propõe uma alternativa para os métodos factíveis usuais ao enfatizar a factibilidade progressivamente com a proximidade da solução (mais especificamente, com a melhoria da aproximação da região factível no passo de minimização e com o ajuste do parâmetro de penalidade). Portanto, restrições muito curvas não provocam encurtamento prematuro dos passos. Outra vantagem deste método é permitir o tratamento de restrições de desigualdade sem a introdução de variáveis de folga. No problema (3.3), que é o problema que desejamos resolver, a inclusão das folgas aumentaria o número de variáveis de $N + 1$ para $m + N + 1$.

Para descrever o método consideramos o problema de programação não linear genérico

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & g(x) \\ \text{sujeito a} \quad & C(x) \leq 0 \end{aligned} \tag{4.1}$$

O algoritmo IR, à semelhança dos procedimentos de programação quadrática seqüencial, divide cada iteração em duas fases: a primeira, de restauração, busca

melhorar a factibilidade; a segunda, de minimização, melhora a otimalidade em relação à função objetivo.

4.1. Restauração

A fase de restauração é executada somente uma vez por iteração e é usada para se obter, a partir do iterando atual x^k, um ponto intermediário y^k que satisfaça às condições

$$\left\|C^+(y^k)\right\| \leq r \left\|C^+(x^k)\right\| \tag{4.1}$$

$$\left\|y^k - x^k\right\| \leq \beta \left\|C^+(x^k)\right\| \tag{4.2}$$

onde $r \in [0, 1)$ e $\beta > 0$ são parâmetros e $C^+(x) = \max\{C_j(x), 0\}$.

4.2. Minimização

A fase de minimização, executada após a fase de restauração, é repetida até que o novo ponto satisfaça os critérios de aceitação baseados na função de mérito. Nesta etapa buscamos reduzir o valor do funcional f exigindo a manutenção da factibilidade até primeira ordem para algumas restrições e respeitando limites da região de confiança.

Para descrever esse passo do algoritmo, vamos definir π_k como uma aproximação linear do subconjunto das restrições "mais ativas" de $C(y^k)$, ou seja,

$$\pi_k = \left\{x \in \mathbb{R}^n | C_j(y^k) + C_j'(x - y^k) \leq C_j^+(y^k) \text{ quando } C_j(y^k) \geq -p\right\} \tag{4.3}$$

Essa redução do número de restrições utilizadas no modelo, baseada em um parâmetro de tolerância p, tem o intuito de diminuir o trabalho computacional por iteração.

Definindo também

$$B_k = \left\{x \in \mathbb{R}^n | \left\|x - y^k\right\| \leq \Delta_k\right\}$$

como a região de confiança centrada no ponto intermediário y^k, o subproblema da fase de minimização pode ser escrito como

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & g(x) \\ \text{sujeito a} \quad & x \in \pi_k \cap B_k \end{aligned}$$

cuja solução denominamos z^k.

4.3. Função de mérito

Métodos que trabalham com pontos infactíveis enfrentam o problema de julgar a qualidade dos iterandos levando em conta, simultaneamente, os objetivos conflitantes de reduzir a infactibilidade e o valor da função objetivo. Uma maneira de

realizar esse julgamento consiste em combinar $g(x)$ e $C(x)$, comparando os pontos com o auxílio das chamadas funções de mérito. Em nosso algoritmo, usamos a seguinte função de mérito

$$\psi(x,\theta) = \theta g(x) + (1-\theta)\left\|C^+(x)\right\|.$$

onde θ é um parâmetro de penalização.

Para decidir a aceitação do ponto z^k, definimos a redução real da função de mérito como a diferença entre o valor da função de mérito no ponto final z^k e no ponto atual x^k, ou seja, $A_{\text{red}} = \psi(x^k,\theta) - \psi(z^k,\theta)$. Definimos também a redução prevista da função de mérito como

$$P_{\text{red}} = \psi(x^k,\theta) - [\theta g(z^k) + (1-\theta)\left\|C^+(y^k)\right\|].$$

Dado um parãmetro $\eta \in (0,1]$, aceitamos o ponto z^k se

$$A_{\text{red}} \geq \eta P_{\text{red}}.$$

Se esta condição não é satisfeita, o ponto candidato é recusado, a região de confiança é reduzida e a fase de minimização é executada novamente.

5. Especialização do método

Para o caso das treliças, as matrizes de rigidez K_i podem ser escritas como produtos diádicos na forma

$$K_i = b_i b_i^T$$

onde b_i são vetores diretamente proporcionais às linhas da matriz de projeção entre o sistema de coordenadas das forças externas e os diversos sistemas de coordenadas alinhados com as barras [3]. Se considerarmos ainda o volume normalizado $V = 1$, o problema (3.3) pode ser reescrito como

$$\begin{aligned} \min_{u\in\mathbb{R}^N,\ \alpha\in\mathbb{R}} \quad & \alpha - f^T u \\ \text{sujeito a} \quad & \frac{1}{2}(b_i^t u)^2 \leq \alpha,\ i = 1,\ldots,m \end{aligned} \tag{5.1}$$

O problema (5.1) é um caso especial da formulação (4.1), com $x = [u,\alpha]^T$ e

$$\begin{aligned} g(x) = g(u,\alpha) &= \alpha - f^T u \\ C_i(x) = C_i(u,\alpha) &= \frac{1}{2}(b_i^T u)^2 \leq \alpha \end{aligned} \tag{5.2}$$

Para definir a fase de restauração, observamos que a variável artificial α é livre nas equações (5.2) e que α aparece em todas as m restrições. Fazendo uso dessa

[3] Essa matriz de projeção é chamada de matriz de compatibilidade. As constantes de proporcionalidade envolvem o comprimento da barra l_i e o módulo de Young E_i

propriedade, propomos um ponto intermediário $\tilde{y}^k = [\tilde{y}^k_u, \tilde{y}^k_\alpha]^T$, sempre factível, obtido com um passo de restauração somente na variável α:

$$\begin{aligned} \tilde{y}^k_u &= u^k \\ \tilde{y}^k_\alpha &= \max_i \frac{1}{2}(b_i^T u^k)^2 \end{aligned} \tag{5.3}$$

Entretanto, como o algoritmo IR pede que o ponto restaurado satisfaça as condições (4.1) e (4.2), é necessária uma escolha criteriosa de α e β ou uma alteração na atualização proposta para que o ponto seja aceito. Somente será possível encontrar um ponto intermediário que satisfaça essas condições e que use um passo na variável α (sem mudar u) se escolhermos os parâmetros β e r satisfazendo

$$\beta + r \geq 1$$

O passo proposto nas equações (5.3) satisfaz a condição (4.1) para qualquer valor de $r \in [0, 1)$ se $\beta \geq 1$. Entretanto se $\beta < 1$ o ponto $\tilde{y}^k$ não vai satisfazer a condição (4.2). Neste caso, se $r < 1 - \beta$, não existirá passo que satisfaça as condições (4.1) e (4.2). Entretanto se $r \geq 1 - \beta$, definindo $y^k = [y^k_u, y^k_\alpha]^T$ conseguimos obter o passo reduzido

$$\begin{aligned} y^k_u &= \tilde{y}^k_u, \\ y^k_\alpha &= \alpha + \beta(\tilde{y}^k_\alpha - \alpha). \end{aligned}$$

Iniciamos a análise da fase de minimização determinando a forma da região factível linearizada descrita na equação (4.3). Seja $z \in \pi_k$, com $z = [z_u, z_\alpha]^T$. Considerando as equações (4.3) e (5.2), concluímos que z deve satisfazer

$$(b_i^T y^k_u)(b_i^T z_u) - z_\alpha \leq \max\left\{\xi - y^k_\alpha, \frac{1}{2}\xi\right\}, \quad i = 1, \ldots, m, \tag{5.4}$$

onde $\xi = (b_i^T y^k_u)^2$.

Assumimos ainda, por simplicidade, que $p > \max|C(x^k)|$, i.e. todas as restrições são consideradas para formar a região π_k. Assim, o subproblema do passo de minimização pode ser escrito na forma

$$\begin{aligned} \min_{z \in \mathbb{R}^{N+1}} \quad & z_\alpha - f^T z_u \\ \text{sujeito a} \quad & \text{a condição (5.4)} \\ & \left\|z - y^k\right\| \leq \Delta_k \end{aligned} \tag{5.5}$$

Este é um problema linear de grande porte, que pode ser resolvido com um emprego pacote computacional apropriado.

6. Alguns resultados numéricos

Para analisar o desempenho do algoritmo proposto, utilizamos o modelo (3.3). Essa formulação foi escolhida por permitir uma série de extensões interessantes como

múltiplos carregamentos, peso próprio da estrutura e reforço. A implementação da base do algoritmo, bem como de um gerador automático de problemas, foi feita em Matlab. O baixo desempenho do código interpretado assim obtido foi contornado parcialmente pela transferência do núcleo de processamento do algoritmo, a fase de minimização, para uma biblioteca de ligação dinâmica compilada[4]. Esse artifício nos permitiu resolver o subproblema (5.5) usando o pacote MINOS compilado em Fortran.

As figuras subseqüentes mostram alguns dos problemas resolvidos. Na legenda de cada figura são indicados o número de variáveis, n, e o número de restrições, m. O número de restrições é igual ao número de barras e, uma vez que as estruturas são planas, o número de variáveis é igual ao dobro do número de nós, menos as restrições dos apoios, mais a variável artificial α. Cada pequeno ponto representa uma rótula em potencial. Linhas pretas representam barras sofrendo compressão e linhas cinza representam barras sofrendo distensão. Os apoios são representados por quadrados quando o nó tem todos os graus de liberdade fixados e por triângulos apontando para a direita quando apenas a componente horizontal está fixa. As setas representam a carga de forças externas aplicada sobre a estrutura. Em todas os casos a estrutura base tem nível de conexão máximo, ou seja, todo par de nós pode ser conectado por uma barra ou uma seqüência de barras colineares.

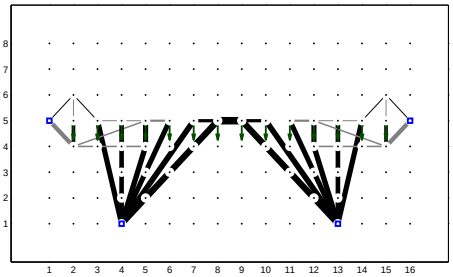

Figura 2: $n = 249$, $m = 3960$.

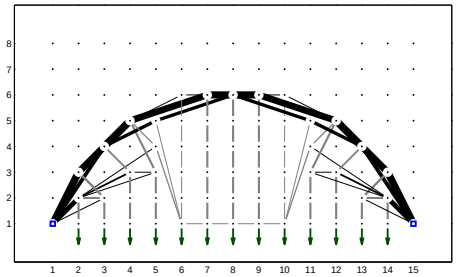

Figura 3: $n = 237$, $m = 3627$.

A Figura 2 ilustra um modelo da vista lateral de uma ponte simples, com quatro pontos de apoio, em dois níveis diferentes. As forças, decorrentes do peso da estrada e dos veículos, são aplicadas ao longo da linha 5. No exemplo mostrado na Figura 3, temos um carregamento distribuído ao longo da linha horizontal que inclui os apoios. Neste caso restringimos o domínio de modo a só permitir que existam barras acima do nível da carga, resultando em uma estrutura familiar em forma de arco.

Na Figura 4, ilustramos a possibilidade de reduzir a dimensão de problemas com simetria especular, como aquele mostrado na Figura 3, que é simétrico em relação ao eixo vertical que passa pela oitava coluna. Neste caso, incluindo uma linha de apoios livres no sentido vertical, obtemos uma boa aproximação para a metade esquerda da estrutura mostrada na Figura 3.

[4]O Matlab chama essas bibliotecas de arquivos MEX e fornece uma API completa para transferência de dados com C ou Fortran

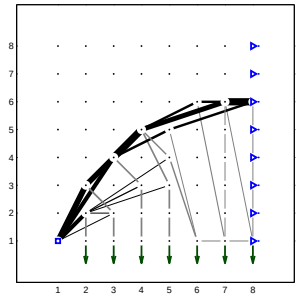

Figura 4: $n = 119$, $m = 1282$.

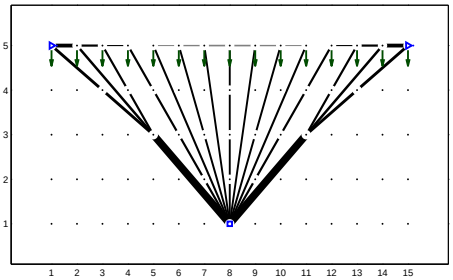

Figura 5: $n = 147$, $m = 1090$.

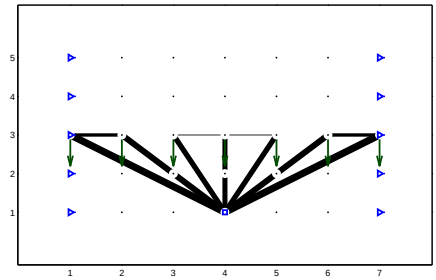

Figura 6: $n = 59$, $m = 386$.

As Figuras 5 e 6 mostram uma extensão do artifício usado na obtenção da Figura 4. Em ambos os casos apenas um apoio fixo é fornecido para suportar uma carga de forças que se extende pela largura da malha. Na primeira figura os apoios foram introduzidos apenas na altura da linha 5. Na segunda figura, permitiu-se a inclusão de apoios em toda a extensão das laterais do domínio. Ainda assim, o projeto final faz uso de apenas dois apoios.

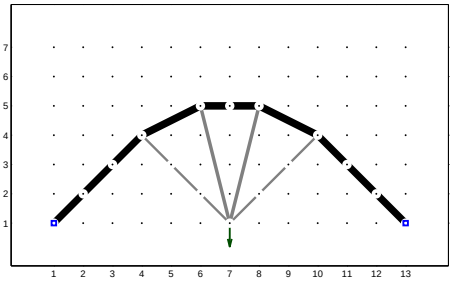

Figura 7: $n = 179$, $m = 2134$.

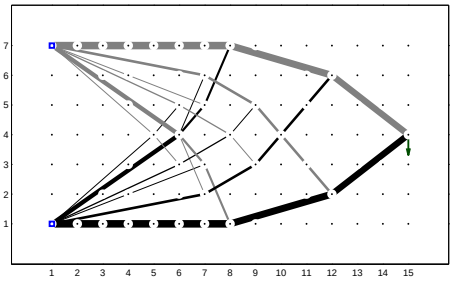

Figura 8: $n = 207$, $m = 3380$.

Finalmente, as Figuras 7 e 8 mostram duas configurações clássicas de solução conhecida, incluídas aqui com o intuito de validar a correção da implementação.

7. Conclusão

Neste trabalho descrevemos em linhas gerais o problema de treliças, a abordagem da estrutura base usada na modelagem e o algoritmo de Restauração Inexata proposto em [3]. Para resolver a reformulação mostrada na equação (3.3) propusemos especializações visando principalmente reduzir o custo computacional de cada iteração.

Para comprovar os resultados obtidos neste trabalho foi implementada também uma versão em Matlab do algoritmo de pontos interiores descrito em [2]. Diversos problemas foram resolvidos usando ambos os métodos e seus resultados comparados quanto à disposição das barras e a rigidez da estrutura. Estes resultados foram sempre muito próximos, raramente discordando na escolha dos elementos. Nos casos em que de fato foram diferentes, observamos que estas diferenças ficavam restritas à redução do volume de algumas barras (normalmente compensada com a introdução de outras mais finas) com pequena influência sobre a rigidez da estrutura final.

Como trabalho futuro, pretendemos implementar em Fortran o algoritmo aqui descrito, pois optamos por não realizar comparações de desempenho usando as versões em Matlab e sim a implementação original do algoritmo proposto em [2], cujo código em Fortran foi gentilmente cedido pelos autores. Entre outros objetivos, pretendemos ainda examinar métodos de identificação de restrições ativas para ajudar na escolha do parâmetro p mostrado na equação (4.3) e implementar o modelo da equação (3.4) que leva em conta múltiplos carregamentos.

Abstract. Trusses are structures formed by thin bars connected by pinned joints, usually loaded only at the joints, presenting, in this case, only axial forces. In this work we determine the optimal geometry of a truss subject to a set of loads, obtaining the stiffest strucure satisfying material and domain constraints. The approach used encompass the sizing and the topological design problems, resulting in a nonlinear programming model. The problem is solved adapting an inexact restoration type algorithm, for which a specialization is proposed taking into account both the model peculiarities and the aim of solving large problems.

Referências

[1] M. P. Bendsøe & O. Sigmund, Topology Optimization, Theory, Methods and Applications, *S*pringer, 4 (2003)

[2] F. Jarre; M. Kočvara; J. Zowe, Optimal truss design by interior point methods, *S*IAM Journal of Optimization Vol. 8, No. 4, pp. 1084-1107, 1998

[3] J. M. Matinez; E. A. Pilota, Inexact - Restoration Algorithm for Constrained Optimization, 1999