UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE TECNOLOGIA E GEOCIÊNCIAS

DEPARTAMENTO DE ENGENHARIA CIVIL E AMBIENTAL

CURSO DE ENGENHARIA CIVIL

JARBAS GONÇALVES DO NASCIMENTO JUNIOR

OTIMIZAÇÃO TOPOLÓGICA EM ESTRUTURAS BIDIMENSIONAIS UTILIZANDO O MÉTODO SIMP

Recife

2023

JARBAS GONÇALVES DO NASCIMENTO JUNIOR

OTIMIZAÇÃO TOPOLÓGICA EM ESTRUTURAS BIDIMENSIONAIS UTILIZANDO O MÉTODO SIMP

Trabalho de conclusão de curso apresentado ao Curso de Engenharia Civil da Universidade Federal de Pernambuco – UFPE, como requisito para obtenção do Grau de Bacharel em Engenharia Civil.

Área de Concentração: Estruturas.

Orientador: Prof. Dr. Renato de Siqueira Motta

Recife

2023

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Nascimento Junior, Jarbas Gonçalves do.
Otimização topológica em estruturas bidimensionais utilizando o método SIMP / Jarbas Gonçalves do Nascimento Junior. - Recife, 2023.
107 p : il.

Orientador(a): Renato de Siqueira Motta
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, Engenharia Civil - Bacharelado, 2023.
Inclui referências, apêndices, anexos.

1. Otimização Topológica. 2. SIMP. 3. MATLAB. 4. Elementos Finitos. I. Motta, Renato de Siqueira. (Orientação). II. Título.

620 CDD (22.ed.)

SERVIÇO PÚBLICO FEDERAL
UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
DEPARTAMENTO DE ENGENHARIA CIVIL E AMBIENTAL
COORDENAÇÃO DO CURSO DE GRADUAÇÃO EM ENGENHARIA CIVIL

ATA DA DEFESA DO TRABALHO DE CONCLUSÃO DE CURSO PARA CONCESSÃO DO GRAU DE ENGENHEIRO CIVIL

CANDIDATO: JARBAS GONÇALVES DO NASCIMENTO JUNIOR

BANCA EXAMINADORA:
Orientador: RENATO DE SIQUEIRA MOTTA
Examinador 1: PAULO MARCELO VIEIRA RIBEIRO
Examinador 2: LEONARDO CORREIA OLIVEIRA

TÍTULO DO TRABALHO DE CONCLUSÃO DE CURSO:
Otimização topológica em estruturas bidimensionais utilizando o método SIMP

LOCAL: RECIFE
DATA: 29 / 05 / 2023
HORÁRIO DE INÍCIO: 14h00**.**

Em sessão pública, após exposição de cerca de 30 minutos, o(s) candidato(s) foi (foram) arguido(s) oralmente pelos membros da banca com NOTA: **9,33** (deixar 'Exame Final', quando for o caso).
1) (X) aprovado(s) (nota > = 7,0), pois foi demonstrado suficiência de conhecimento e capacidade de sistematização no tema da monografia e o texto do trabalho aceito.
As revisões observadas pela banca examinadora deverão ser corrigidas e verificadas pelo orientador no prazo máximo de 30 dias (o verso da folha da ata poderá ser utilizado para pontuar revisões).
O trabalho com nota no seguinte intervalo, **3,0 = < nota < 7,0,** será reapresentado, gerando-se uma nova ata; sendo o trabalho aprovado na reapresentação, o aluno será considerado **aprovado com exame final**.
2) ( ) reprovado(s). (nota <3,0)

Na forma regulamentar foi lavrada a presente ata que é assinada pelos membros da banca e pelo(s) candidato(s).

Recife, 29 de maio de 2023

Orientador – Renato de Siqueira Motta: .....

Avaliador 1 – Leonardo Correia Oliveira: .....

Avaliador 2 – Paulo Marcelo Vieira Ribeiro: .....

Candidato 1 – Jarbas Gonçalves do Nascimento Junior: .....

AGRADECIMENTOS

À Deus, por toda a graça derramada em minha vida, por me fornecer forças, fé, coragem e sabedoria para prosseguir e conseguir chegar ao final deste ciclo

À minha mãe Gildete, por ter sempre confiado em mim e nas minha escolhas, por sempre ter estado ao meu lado, me apoiando, por ter me fornecido todas as chances possíveis para eu me tornar a melhor pessoa. Seu amor, carinho e cuidado comigo nunca serão esquecidos.

À minha amada esposa Andressa, por ter passado por todas as fases da engenharia junto comigo, desde o curso técnico em edificações até hoje. Por ter sempre me apoiado na busca dos meus sonhos e por me aturar sempre que algo não agradável acontecia. Por sempre me encorajar a buscar cada vez mais conhecimento. Te amo!

Ao professor Renato Motta, por ter me apresentado um tema tão interessante e inovador para o desenvolvimento deste trabalho.

RESUMO

Neste trabalho, foi utilizado um método de otimização topológica (MOT), destinado a minimizar a flexibilidade com o volume fixo. O MOT utilizado foi o SIMP (Solid Isotropic material with Penalization), que tenta penalizar concentrações intermediárias considerando apenas concentrações com valor 0 ou 1, então o resultado é bem definido e alcançável. Ao desenvolver o modelo o método dos elementos finitos (MEF) é usado para distinguir áreas e avaliar funções objetivo e restrições postas em problemas de otimização. Este método não depende da geometria inicial, que produz uma geometria de alta complexidade. Para obter geometrias claras e fabricáveis, é necessário refinar a malha de elementos finitos, o que pode trazer problemas numéricos como a dependência da malha e o aparecimento de zonas com densidades intermediárias, também chamadas de tabuleiros de xadrez. Para evitar problemas numéricos, filtros de densidade e sensibilidade são implementados. As simulações realizadas, foi utilizado um modelo desenvolvido por Andreassen e Clausen, *et al*. algoritmo didático proposto em 2009 na linguagem MATLAB. Foram selecionados casos do cotidiano de engenheiros civis, como por exemplo vigas, pontes e edifícios. Os resultados obtidos foram avaliados em termos de eficiência de filtragem, tempo de convergência do código, ocorrência de instabilidade numérica e uma simples comparação com os resultados publicados na literatura.

Palavras-Chave:

- Otimização Topológica, Método dos Elementos Finitos, Viga, Prédio, MATLAB, SIMP

ABSTRACT

In this work, a topological optimization method (MOT) was used, intended to minimize the flexibility with the fixed volume. The MOT used was SIMP (Solid Isotropic Material with Penalization), which tries to penalize intermediate concentrations by considering only concentrations with value 0 or 1, so the result is well-defined and achievable. When developing the model, the finite element method (FEM) is used to distinguish areas and evaluate objective functions and constraints posed in optimization problems. This method does not depend on the initial geometry, which produces a complex geometry. To obtain clear and manufacture geometries, it is necessary to refine the finite element mesh, which can bring numerical problems such as mesh dependence and the appearance of zones with intermediate densities, also called checkerboards. To avoid numerical problems, density and sensitivity filters are implemented. For the simulations performed, a model developed by Andreassen and Clausen, *et al*. was used. the didactic algorithm proposed in 2009 in MATLAB language. Cases from the everyday life of civil engineers, such as beams, bridges, and buildings, were selected. The results obtained were evaluated in terms of filtering efficiency, code convergence time, the occurrence of numerical instability, and simple comparison with results published in the literature.

Keywords:

- Topological Optimization, Finite Element Method, Beam, Building, MATLAB, SIMP

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIAÇÕES

Sigla	Descrição
2D	Bidimensional
3D	Tridimensional
AM	Manufatura Aditiva (Additive Manufacturing)
E	Módulo de elasticidade de Young
E_0	Módulo de elasticidade de Young inicial
E_k	Módulo de elasticidade de Young do elemento
E_{min}	Módulo de elasticidade de Young mínimo
GL	Grau de Liberdade
IDE	Ambiente de desenvolvimento integrado (*Integrated Development Environment*)
KKT	Karush-Kuhn-Tucker
MEF	Método de Elementos Finitos
MOT	Método de Otimização Topológica
CO	Critério de Ótimo
OT	Otimização Topológica
r_{min}	Raio mínimo do elemento finito
SIMP	Material Isotrópico Sólido com Penalização (*Solid Isotropic Material with Penalization*).

LISTA DE SÍMBOLOS

Símbolo	Descrição
Alfabeto latino	
B_k	Fator multiplicador
I_s	Matriz identidade
d_k	Direção da descida
$\bar{u}$	Vetor de deslocamento nodais
$\nabla f(x^*)$	Gradiente da função f(x)
F	Vetor de carregamento
f_p	Número total de funções objetivo
H	Fator peso utilizado nos filtros
H(x)	Matriz Hessiana
H^1	Espaço de Hilbert
k	Matriz de rigidez auxiliar
K	Matriz de rigidez global
Ke	Matriz de rigidez elementar
M^e	Matriz de massa elementar
N	Matriz Linha das funções de interpolação ou função de forma
nelx	Número de elementos da malha na direção horizontal
nely	Número de elementos da malha na direção vertical
nf	Número total de funções objetivo
ng	Número de restrições de desigualdade
nh	Número de restrições de igualdade
nvp	Número de variáveis do problema

nvp	Número de variáveis do problema
x̃	Vetor de variáveis do problema
c	Flexibilidade
ch	Condição de parada
x	Densidade do elemento
$\tilde{x}$	Densidade artificial do elemento

Alfabeto grego

Γ_D	Contorno do domínio
α	Passo
η	Coeficiente de penalização
λ	Multiplicador de Lagrange
Ω	Domínio
$\boldsymbol{\nu}$	Coeficiente de Poisson
δ	Coeficiente de amortecimento

Sumário

1. INTRODUÇÃO

Com a busca da melhor relação custo-benefício em todas as áreas da engenharia é necessário utilizar algumas alternativas para chegar a melhor relação. Assim é necessário buscar todas as propriedades que o material pode fornecer considerando os esforços que ele está sendo solicitado. Tendo essa necessidade, foi criado o conceito de Otimização Topológica (OT), sendo esse campo de pesquisa da engenharia focado em otimizar a topologia das estruturas, ou seja, dar uma nova forma e geometria das estruturas, segundo critérios pré-estabelecidos, maximizando ou minimizando a variável desejada. Esses critérios podem ser restrições de movimentação da peça, minimizar as deformações nas estruturas, maximizar rigidez, minimizar a massa de acordo com a distribuição de tensões da estrutura, minimizar o volume do material, entre outros. Segundo M. P. BENDSOE e O. SIGMUND (2003), essa distribuição ótima de material consiste em se determinar em quais pontos do domínio conterão material isotrópico e quais pontos estarão vazios. Pois os locais com baixa ou nenhuma solicitação podem ser retirados da estrutura, deixando vazios que definem a nova forma de estrutura.

Os conceitos que dão suporte teórico ao método estão estabelecidos há várias décadas, pois é utilizado o Método de Elementos Finitos (MEF) para avaliar o desempenho da estrutura, além da utilização de técnicas de programação matemáticas baseadas no gradiente como o grande fator para otimização do projeto.

1.1 MOTIVAÇÃO

No mundo a busca pelo ótimo acontece em todas as áreas, em relação à engenharia estrutural há a busca pela estrutura que suporte os mesmos esforços e que seja necessário o mínimo de material. É nesse contexto que a OT é inserida e é de grande valia, já que com a altos do preço dos insumos, a busca pela sustentabilidade, e a limitada obtenção de matéria-prima encarece bastante os tradicionais métodos construtivos aplicados no Brasil, com isso a OT aplicada como redutor de volume das estruturas faz com que diminua o quanto de material é necessário para aquela estrutura ser funcional para o que foi prevista.

Além disso, a OT pode ser inteiramente ligada as novas tecnologias que estão sendo desenvolvidas na construção civil, como, por exemplo, a impressão de peças em 3D, pois assim é possível modelar o elemento estrutural computacionalmente e após otimizado só é necessário imprimir. Mas isso em relação ao Brasil ainda está um pouco distante, dado que há poucos estudos aplicando otimização topológica em projetos de engenharia estrutural. Porém, esse é o futuro da engenharia, já que facilitaria a obtenção de estruturas com menor massa, não sendo necessário a obtenção de fôrmas e o elemento podendo ter qualquer forma, contanto que ele suporte os esforços solicitantes e seja funcional.

Com isso é de suma importância o desenvolvimento de trabalhos em relação a OT, pois como foi dito é o futuro da engenharia estrutural e consequentemente da construção civil.

1.2 OBJETIVOS

1.2.1 Objetivo geral:

O presente trabalho tem como objetivo geral o desenvolvimento teórico-experimental de otimização topológica em estruturas bidimensionais com auxílio de ferramentas computacionais, buscando atender as soluções estruturais propostas a partir da redução de volumétrica da peça em análise.

Visa também entender como funciona a lógica de um código computacional para otimização topológica utilizando o método SIMP (*Solid Isotropic Material with Penalization)*.

1.2.2 Objetivos específicos:

- Compreender os conceitos de Otimização Topológica;
- Analisar códigos de otimização Topológica para a estrutura de interesse;
- Analisar a eficiência dos filtros de sensibilidade e densidade do trabalho de Andreassen e Clausen, *et al*;

- Realizar simulações numéricas das estruturas de interesse;
- Realizar análise comparativa entre códigos de otimização topológica;
- Realizar análise entre os resultados obtidos e os presentes na literatura.

2. REVISÃO BIBLIOGRÁFICA

2.1 Histórico

A seguir será descrito alguns marcos importantes para a fomentação e criação da otimização topológica e será feito um breve resumo do que cada uma das personalidades contribuiu para o desenvolvimento do campo de estudo.

2.1.1 James Maxwell

A otimização estrutural é um assunto de interesse há mais de um século. Um dos primeiros estudos sobre o assunto foi feito por James Maxwell em 1870, onde ele buscou formular teoremas relativos a racionalizar o design estrutural, o trabalho tinha com ênfase estruturas de pontes, obras de artes de grande interesse para a engenharia. Maxwell, no seu estudo, aplicou a teoria de elasticidade para problemas simples, onde ele buscava o risco de falha dos elementos conforme ele iria retirando a quantidade de material e com isso ele buscava a massa ótima que suportaria a carga esperada com o mínimo de massa.

O estudo que Maxwell desenvolveu, era na sua essência, um carregamento atuando num domínio e nos pontos onde esse ponto estaria apoiado, calcular o campo de tensões mecânicas principais usando a teoria de elasticidade. Ele verificou a direção das tensões principais e constatou que essa direção é onde não ocorrem as tensões de cisalhamento, sendo atuante apenas as tensões normais. Com essas direções, Maxwell pode sugerir uma forma conceitual de otimização da estrutura utilizando menos material seria constituída de elementos de treliças, com as barras alinhadas com essas direções das tensões principais. Essa solução, algum tempo depois, veio ser reconhecida como solução ótima, dado um carregamento único, para a estrutura com menor peso, máxima rigidez, reduzindo a quantidade de material necessária e, consequentemente, o custo de produção.

2.1.2 Anthony Michell

Em 1904, Anthony Michell continuou com o trabalho de Maxwell e ainda hoje as suas estruturas são referências na teoria de otimização topológica. Michell decidiu aplicar o método para um projeto de vários tipos de estruturas utilizando o mesmo critério de máxima rigidez com menor volume do material. Ele chegou a elementos treliçados. Porém, seu estudo foi além do que Maxwell propôs, o estudo de Michell tem como princípio calcular o campo das principais isotensões. Ele utilizou a teoria de elasticidade no caso de uma força aplicada a um ponto do domínio sujeito a restrições de deslocamento nos outros pontos. Com isso ele obteve as linhas de isotensões principais e assim alinha as barras nas mesmas direções das tensões principais calculas do domínio. Sendo assim, a solução ótima seria a que os elementos estariam sujeitos apenas a tração e compressão, não existindo momento fletor.

Contudo, os resultados obtidos por Michell foram considerados acadêmicos demais e sem aplicabilidade, pois eram muito difíceis de serem construídos no início do século XX. Somente na década de 80, com a utilização de softwares baseados em otimização topológica que os resultados de Michell passaram a ser reproduzidos.

2.1.3 Martin Philip Bendsoe e Noboru Kikuchi

As aplicações para mecânica estrutural e otimização topológica tornou-se reconhecidas como um campo de estudo após as publicações de Martim Philip Bendsoe e Noboru Kikuchi em 1988. A partir deste momento a aplicação da sistemática do MEF e de programação matemática utilizada para problemas de otimização não-linear com restrições de desigualdade para projetos de treliças e barras foi mais bem compreendido e foi verificado que era uma solução viável.

2.1.4 Breve histórico de contribuições recentes

Após as contribuições de Bendsoe e Kikushi a quantidade de estudos produzidos aumentou muito, juntamente com o aumento da quantidade de linguagens computacionais, como Python e Fortran, mas a que mais se destacou foi o MATLAB.

Surgiram códigos buscando otimizar ainda mais o código de Sigmund (2001) que continha 99 linhas, como pode ser visto nos trabalhos de Andreassen e Clausen, *et al.* (2009) que propôs um código com 88 linhas. Porém, como esses códigos representavam estruturas em 2D, surgiram trabalhos que fornecessem estruturas otimizadas em 3D como os de Ferrari (2020) e de Liu e Tovar (2014) que buscaram diminuir a utilização de processamento, sem diminuir a qualidade do resultado.

Além disso, vários softwares de análise estrutural implementaram dentro de suas rotinas a possibilidade de utilização de otimizar a estrutura topologicamente, um exemplo é o software ANSYS ®.

Diariamente novidades aparecem no mundo da computação e um dos temas mais discutidos é a utilização de redes neurais, que se trata de ensinar ao computador como deve fazer para realizar certa tarefa. Está metodologia já está sendo empregada também para o desenvolvimento de estruturas ótimas. São poucas pesquisas relacionadas ao assunto, mas temos o exemplo da pesquisa de Sosnovik e Oseledets (2019) que é a que mais difundida.

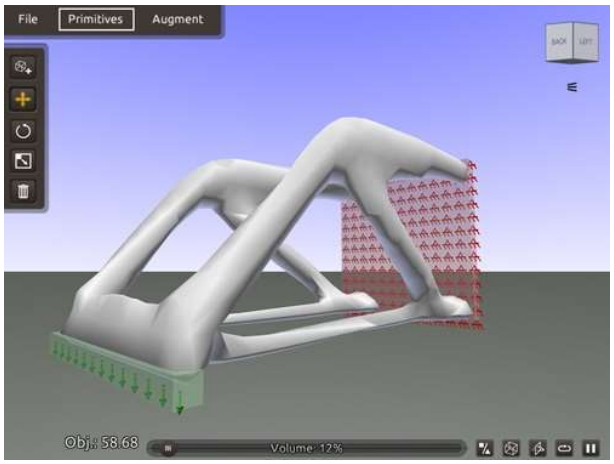
Além de estudos e simulações, várias são as utilizações da otimização topológica, como na área de mecanismos mecânicos, de aeronáutica, da acústica e da construção civil, como por exemplo a construção do Centro de Convenções Nacional do Qatar que utilizou uma estrutura otimizada topologicamente (Figura 1). Onde o projetista Arata Isozaki utilizou um modelo otimizado topologicamente para servir de coberta da fachada.

FIGURA 1 – ESTRUTURA DE FACHADA DO CENTRO DE CONVENÇÕES DO QATAR.

Fonte: Nelson Garrido, 2011

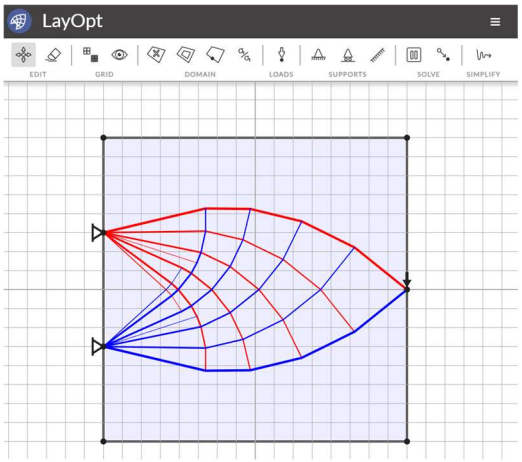
Além de códigos de linguagem computacional, há aplicativos para celulares e computadores que pode realizar simulações de otimização topológica, como, por exemplo, o *TopOpt (*Figura 2*)*, *nTop*, *LayOpt (*Figura 3*),* entre vários outros que já disponibilizam uma interface intuitiva para uma obtenção de uma estrutura ótima de forma rápida. Inclusive, o tempo de resposta dos aplicativos é melhor do que dos códigos desenvolvidos pelo MATLAB, isto se dá por todo estrutura computacional envolvida para o desenvolvimento de um aplicativo e pela linguagem utilizada no desenvolvimento do programa.

FIGURA 2 – ESTRUTURA OTIMIZADA NO *TOPOPT*



Fonte: Niels Aage, 2014

FIGURA 3 – ESTRUTURA OTIMIZADA NO *LAYOPT*



Fonte: Autor, 2023

2.2 Premissas para o desenvolvimento do código

Para o desenvolvimento de qualquer código é necessária uma lógica por trás, com isso nessa seção será desenvolvido o que é necessário para desenvolver um código do método de otimização topológica.

2.2.1 Otimização estrutural

2.2.1.1 *Conceito*

Uma etapa essencial para a solução de problemas de otimização estrutural consiste na solução de equações de equilíbrio, podendo ser analítica ou numericamente. Comumente as variáveis de projeto, a função objetivo e as restrições dependem direta ou indiretamente da solução das equações de equilíbrio. Por isso, a correta solução dessas equações é fundamental na busca pelo ótimo.

Neste trabalho será utilizado as equações da Teoria da Elasticidade Linear Infinitesimal Isotrópica para o cálculo da resposta estrutural do sistema. Essa escolha foi realizada com base no problema que se deseja resolver: todas as estruturas analisadas apresentarão uma relação entre tensão e deformação dentro do regime elástico, não interessando fenômenos relacionados à não-linearidade física e geométrica como plastificação e fratura, que podem ocorrer principalmente na região de aplicação das cargas e deslocamentos prescritos.

Nessa abordagem infinitesimal, a otimização estrutural, pode ser classificada em três grupos de otimização: paramétrica, de forma e topológica.

Sant’Anna (2002) fala que na otimização paramétrica é realizado a alteração direta das variáveis do projeto como diâmetro, espessura, entre outras. Nesse grupo uma vez que é escolhida uma geometria ela não é alterada no processo de otimização.

A otimização de forma, permite a alteração da geometria dos componentes, mas não é permitido a alteração da posição desses elementos. É um modelo mais antigo do que a paramétrica.

A otimização topológica, foco deste trabalho, onde o domínio é fixo, definido e não se modifica durante o processo, mas para atingir a solução ótima é realizado

uma distribuição de material no domínio. Essa distribuição pode ser feita com a inserção de vazios ou adição de material. O resultado dessa distribuição pode gerar estruturas com geometrias convencionais ou não convencionais.

O otimização visa obter os máximos ou mínimos das funções. Aplicando o MEF, os valores acabam sendo vetorizados, com isso é necessário definir um modelo de otimização, quais as variáveis de projeto, qual o objetivo, restrições imposta no problema e qual a ferramenta matemática que será utilizada na busca do ótimo.

2.2.1.2 *Função objetivo e restrições*

Sant'Anna (2002) diz que a busca pelo ótimo não apresenta uma única solução possível, como o modelo é interativo pode apresentar diversas soluções possíveis que atendem os objetivos e as restrições. Porém, algumas soluções são melhores do que outras, mas nem sempre a melhor solução é viável devido a questões operacionais de programação e de fabricação. Com isso é necessário estabelecer qual é a função objetivo, as restrições e o critério de parada utilizado para solução.

Após estabelecer a função objetivo, define-se as restrições com todos os requerimentos impostos às variáveis de projeto. Alguns exemplos são: as restrições de tensão e flexibilidade. Tais restrições podem ser de igualdade ou desigualdade. As restrições de igualdade estão presentes em problemas de otimização estrutural nas equações de equilíbrio e nas restrições de volume.

Selecionar a função objetivo é muito importante no processo, assim como a seleção de variáveis de projeto, as restrições e o uso de hipóteses simplificadores. Neste trabalho, a função a ser minimizada é a flexibilidade e o volume é considerado uma restrição.

Segundo Sant'Anna (2002), a redução do volume e minimização da flexibilidade resulta em estruturas mais compactas e diminui o desperdício de material, sendo um ponto positivo com relação ao financeiro e a sustentabilidade. Porém, nem sempre a redução dos desperdícios reflete numa redução de custo, pois dependendo da complexidade da solução achada pode causar um custo superior a uma peça

original. Este trabalho não considerou os custos de fabricação na função objetivo, mas sim buscou obter a topologia ótima do elemento.

2.2.1.3 *Vetor Gradiente e Hessiana*

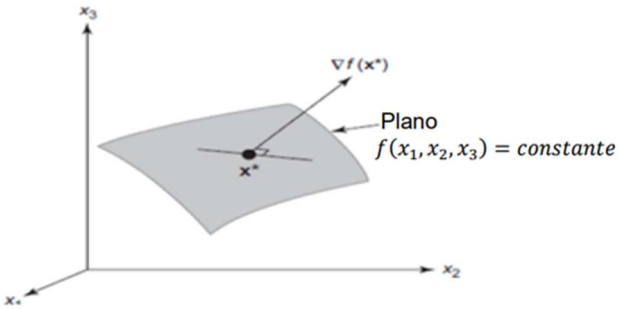
A matriz hessiana e o vetor gradiente são elementos do cálculo que auxiliam na percepção gráfica e matemática da solução de um problema de otimização, por isso que merecem serem detalhados.

O vetor gradiente, é o vetor de derivadas da função objetivo em relação a variável de projeto, esse vetor determina a sensibilidade de busca pelos extremos (x^*).

$$\nabla f(x^*) = \begin{bmatrix} \dfrac{\partial f(x^*)}{\partial x_1} \\ \dfrac{\partial f(x^*)}{\partial x_2} \\ \vdots \\ \dfrac{\partial f(x^*)}{\partial x_n} \end{bmatrix}$$

O vetor gradiente é definido como sendo o vetor normal ao plano tangente no ponto x*. Nesse ponto o vetor gradiente indica a direção e sentido na qual ocorrerá maior taxa de variação da função objetivo, isto é, aponta para os máximos ou mínimos. Com isso, para determinar o ponto mínimo basta utilizar a direção oposta, isto é, $-\nabla f(x^*)$. A Figura 4 foi apresentada no trabalho de Nascimento (2019) e representa bem o que seria o vetor gradiente para uma função de três variáveis em um plano conhecido.

FIGURA 4 – GRADIENTE DE UMA FUNÇÃO.



Fonte: Nascimento, 2019.

A hessiana, é o outro componente de cálculo que deve ser determinado para minimizar as funções. Ela é a combinação de derivadas mistas de segunda ordem do vetor gradiente, como pode ser visto na expressão abaixo:

$$H\ (x^*) = \begin{bmatrix} \frac{\partial f(x^*)}{\partial {x_1}^2} & \frac{\partial f(x^*)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial f(x^*)}{\partial x_1 \partial x_n} \\ \frac{\partial f(x^*)}{\partial x_2 \partial x_1} & \frac{\partial f(x^*)}{\partial {x_2}^2} & \cdots & \frac{\partial f(x^*)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f(x^*)}{\partial x_n \partial x_1} & \frac{\partial f(x^*)}{\partial x_n \partial x_2} & \cdots & \frac{\partial f(x^*)}{\partial {x_n}^2} \end{bmatrix}$$

A hessiana, também tem como função achar os máximos e mínimos da função objetivo quando ela tem várias variáveis.

2.2.1.4 *Teorema do ponto fixo*

Geralmente, os algoritmos usados para minimizar ou maximizar uma função objetivo utilizam do teorema do ponto fixo concomitantemente com a regra da descida. Segundo Arora (2012), o teorema do ponto fixo pode ser expresso conforme a equação abaixo:

$$x_{k+1} = x_k + \alpha_k d_k$$

O x representa uma variável de projeto, α_k representa o valor do passo e d_k é a direção da descida. Com isso, o problema de otimização é divido em duas partes, a busca pela direção da otimização e o quanto é necessário descer para conseguir o ótimo, o que caracteriza o passo.

O passo e a direção, pode ser avaliado de diversos métodos analíticos ou heurístico podendo ser abordado por programação linear ou não linear. Nesse trabalho, foi utilizado um método baseado em programação não linear que utiliza um esquema de atualização heurístico oriundo do trabalho de Andreassen e Clausen (2010).

2.2.2 Método Otimização topológica (OT)

Derivada das palavras gregas topos, “lugar” e logos, “estudo”, a otimização topológica é um método computacional de distribuição de material que busca analisar estruturas sem forma preestabelecida, dando ao método liberdade para que seja encontrado design estruturais esbeltos e de alta confiabilidade, devido as suas simulações.

A otimização topológica é uma área de estudo que busca a melhor distribuição do material da estrutura apoiada e carregada, o que é chamado condição de contorno e para a isto é necessário a utilização do MEF e as fórmulas matemáticas de otimização.

Em 1988, Bendsoe e Kikuchi propuseram a otimização da forma e eles consideraram a equação constitutiva homogeneizada que dependem da densidade relativa do material. Ou seja, o algoritmo criado modifica os elementos finitos variando a densidade de cada elemento tornando a densidade de cada elemento desse em binário, pois ele ou é 0 ou é 1. Sendo nulo, não há material preenchendo o espaço elemento finito, sendo 1 há material preenchendo todo o elemento finito e sendo valores entre 0 e 1 há material em algumas áreas do elemento finito.

Mas o estudo ainda não estava finalizado, já que havia espaços que estavam sendo preenchidos com qualquer valor diferente de 0 e 1, com isso pesquisas posteriores foram aperfeiçoando os algoritmos para que fosse determinado a partir de qual valor entre 0 e 1 deveria ser considerado o espaço vazio ou preenchido. Assim teria uma estrutura com geometria e massa otimizada. Um dos métodos que utiliza essa abordagem é o método SIMP (*Solid Isotropic Material with Penalization*) e tal método será utilizado neste trabalho.

2.2.2.1 Material Isotrópico Sólido Com Penalização (SIMP)

A sigla SIMP vem do inglês *Solid Isotropic Material with Penalization* que quer dizer material sólido isotrópico com penalização. Este método propõe a distribuição de material com microestruturas artificiais para solucionar o problema do

ótimo. O método foi inicialmente proposto por Bendsoe em 1989 e modificado por Andreassen em 2010.

O método prediz a melhor distribuição do material em um domínio definido, para determinado caso de carga, condições de contorno, restrições e requisitos de desempenho. Tradicionalmente a otimização topológica é feita com a discretização do domínio em uma grade de elementos finitos chamados de malha. Segundo Bendsoe: A otimização topológica em sua definição mais geral deve consistir na determinação da existência ou não de material em cada elemento do espaço discretizado.

Cada elementos da malha de elementos finitos é preenchido com material para as regiões que precisam de material ou é retirado quando não precisa de material. Como falado anteriormente, a distribuição de densidades do material, ρ, é distinta e cada elemento recebe um valor binário, sendo o valor 1 para o material preenchido e 0 para a ausência de material.

Com isso o processo de otimização pelo método SIMP modificado é definido pela equação:

$$E_k(x_k) = E_{min} + x_k^{\eta}(E_0 - E_{min}), x_k \in [0,1]$$

onde η é o coeficiente de penalização, cujo valor deve ser maior ou igual a 1, já que ele serve para penalizar as densidades intermediárias. E_k e E_0 são os módulos de Young do elemento e do material respectivamente. E_{min} é o valor mínimo que o módulo de Young pode assumir, evitando assim singularidades na matriz de rigidez. x é a densidade do elemento.

2.2.3 Definições matemáticas para otimização

Para descrever os conceitos e método de otimização é necessário realizar uma definição do formato matemático que o problema apresenta. Desse modo o modelo de otimização padrão é expresso da seguinte forma:

Encontra-se um vetor de variáveis de projeto $\tilde{X} = \tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_{nvp}$, com *nvp* sendo o número total de variáveis do problema. Com isso, é buscado minimizar a função objetivo:

$$f_p(\tilde{x}) = f_p(\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_{nvp}), sendo\ p\ o\ objetivo$$

estando sujeito a *nh* restrições de igualdade,

$$h_j(\tilde{x}) = h_j(\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_{nvp}) = 0, sendo\ j\ de\ 1\ até\ nh$$

e estando sujeito a *ng* restrições de desigualdade,

$$g_i(\tilde{x}) = g_i(\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_{nvp}) \leq 0, sendo\ i\ de\ 1\ até\ ng$$

Com o conjunto de variáveis do problema satisfazendo todas essas restrições, ela é considerada um conjunto viável, caso ela não se enquadre em alguma dessas restrições o problema é considerado inviável. Tais condições foram primariamente publicadas de forma simplificada por William Karush em 1939, mas só se tornaram conhecidas quando publicadas por Harold W. Kuhn e Albert W. Tucker em 1951 e é conhecido como método Karush-Kuhn-Tucker (KKT).

Com isso, as condições necessárias para satisfazer o KKT:

I. Condição necessária: Para que x^* seja um extremo local da função $f(x)$, diferenciável em x^* é necessário que o gradiente da função seja zero:

$$\nabla f(x^*) = 0$$

II. Condição necessária de 2° ordem: Para que x^* seja um extremo local da função $f(x)$, duas vezes diferenciável em x^*, $H(x^*)$ não pode ser zero e deve ser semi-definida.

Com a hessiana ($H(x^*)$) sendo positiva ou negativa semi-definida é possível pressupor que todos os seus autovalores são maiores ou iguais a zero para que x^* seja um ponto de mínimo local.

III. Condição suficiente: Sendo a função $f(x^*)$ duas vezes diferenciável em x^* de forma que $H(x^*)$ é diferente de zero e definida, garante que x^*é um ponto de extremo.

IV. Condição de Complementariedade: O produto entre o multiplicador de Lagrange e a função objetivo deve ser zero.

$$\lambda_i g_i(x^*) = 0$$

2.3 Método dos elementos finitos (MEF)

O método de elementos finitos, é um método numérico utilizado para simular um meio contínuo. Ele vem sendo amplamente utilizado para simular o comportamento de estruturas segundo a mecânica dos sólidos deformáveis, uma vez que ele torna possível a modelagem de estruturas com as mais diversas geometrias, carregamentos e condições de contorno. Na engenharia moderna é muito difícil que um problema não precise utilizar alguma ferramenta de cálculo que envolva o MEF.

O MEF tem como princípio encontrar uma solução para um problema complicado por meio de substituição desse problema por um mais simples. É aproximado a solução de um modelo contínuo pela solução de um modelo discreto, isto é, tomando um problema com valor e contorno inicial, cujo comportamento é descrito por um sistema de equações diferenciais e substituindo-o por um problema de equações algébricas, o que permite descrever o comportamento da estrutura de forma aproximada, tornando possível a solução de problemas complexos.

Segundo Cook, *et al*. (2002), para utilizar o MEF é necessário seguir alguns passos:

I. Dividir a estrutura em elementos finitos, gerando assim uma malha;

II. Formular as propriedades de cada elemento;

III. Realizar a montagem das matrizes e vetores globais da estrutura;

IV. Aplicar os carregamentos prescritos;

V. Especificar os deslocamentos nodais prescritos;

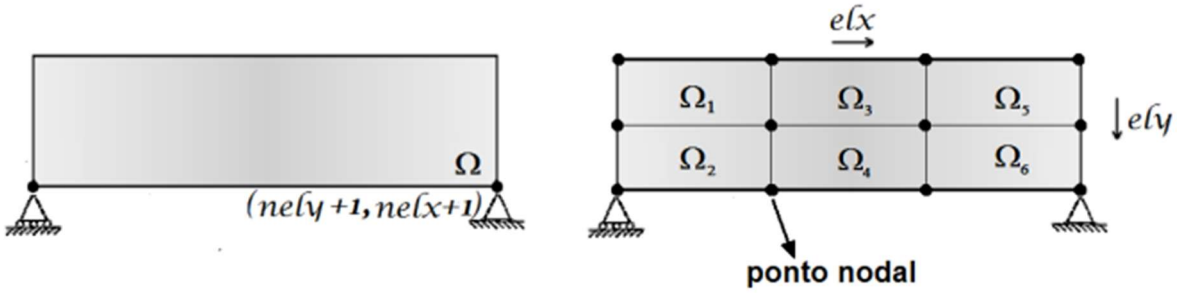
VI. Resolver simultaneamente as equações algébricas lineares para determinar os deslocamentos nodais;

VII. Calcular as deformações e tensões do elemento.

Como pode ser visto na Figura 5, é realizado a discretização de uma viga utilizando o MEF. Do lado esquerdo temos a representação de uma viga biapoiada no

domínio Ω, com as condições mecânicas e geométricas definidas, sendo assim um problema contínuo. A direita, é possível ver o problema discretizado em uma malha com 6 elementos, interligados por pontos nodais localizados nos seus vértices.

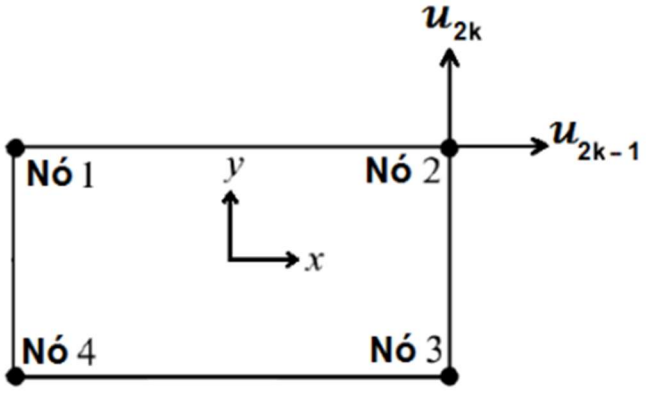
FIGURA 5 – DISCRETIZAÇÃO DE UMA VIGA UTILIZANDO O MEF.



Fonte: Maia, 2021.

É considerado um sistema de coordenadas cartesianas (x, y) para cada elemento finito empregado na discretização, tendo como origem seu centroide e os quatro pontos nodais posicionados em seus vértices, numerados no sentido horário de 1 a 4. Cada ponto nodal possui coordenadas que se referem as coordenadas locais do elemento e o deslocamento nodal u_{2k-1} e u_{2k} nas direções de x e y respectivamente. Com isso, cada elemento finito possui 8 graus de liberdade, como pode ser visto na Figura 6.

FIGURA 6 – ELEMENTO FINITO RETANGULAR COM 4 PONTOS NODAIS E 8 GL.



Fonte: Maia, 2021.

2.3.1 Definições matemáticas para utilização do MEF

O método dos elementos finitos é um método numérico utilizado como ferramenta para obtenção de soluções aproximadas para equações diferenciais parciais. E assim, os problemas regidos por estas equações, com algumas exceções, não possuem solução analítica conhecida e com isso o MEF se faz importante, pois ele busca tais soluções.

De acordo com Reddy (2006), O MEF se baseia na divisão do domínio do problema em subdomínios, que são denominados elementos finitos, sobre os quais a equação diferencial dominante é aproximada utilizando um método variacional. Para problemas da mecânica estrutural, a forma variacional por ser escrita como (REDDY, 2006):

Encontrar a função $u \in \overline{U}(\Omega)$, tal que:

$$B(u,q) = F(q), \forall\, q \in \bar{S}(\Omega)$$

onde $B(.,.) : \overline{U}\, x\, \overline{S} \rightarrow R$ é uma função bilinear contínua e simétrica, $F(.): \bar{S} \rightarrow R$ é um funcional contínuo e linear, e q é uma função de variação genérica e cinematicamente admissível. Para isso $\overline{U}$ e $\overline{S}$ são espaços de função, definidos no domínio de análise $\Omega \subset R^d$ (onde d representa a dimensão física do problema) e é dado respectivamente por:

$$\overline{U} = \overline{U}(\Omega) = \{u \in H^1(\Omega)\ | u = \bar{u}\ em\ \Gamma_D\}$$

e

$$\bar{Q} = \bar{Q}(\Omega) = \{u \in H^1(\Omega)\ | u = 0\ em\ \Gamma_D\}$$

onde H^1 é o espaço de Hilbert de primeira ordem e Γ_D é o contorno do domínio com condições de contorno de Dirichlet prescritas.

É necessário frisar que o problema variacional apresenta uma única solução e é comprovado através do Teorema generalizado de Lax-Milgram e sucede de características apresentadas na forma bilinear *B (.,.)*, como continuidade e coercividade.

Considerando um problema variacional aproximado e definido dentro de um subespaço de funções de aproximação $U^{MEF} \subset \overline{U}$, com isso é obtido uma função aproximada de elementos finitos u^{MEF}, que substitui a função u. No novo domínio, discretizado, o problema variacional pode ser posto como:

Encontrar $u^{MEF} \in U^{MEF}$, tal que:

$$B(u^{MEF}, q^{MEF}) = F(q^{MEF}), \forall\, q^{MEF} \in Q^{MEF}(\Omega) \subset \bar{Q}(\Omega)$$

onde, U^{MEF} e V^{MEF} são, respectivamente, os subespaços de função polinomial por parte de ordem p, contidos em $\overline{U}$ e $\bar{Q}$.

A união desses elementos finitos é dada o nome de malha de elementos finitos. Neste caso, um sistema matricial de equações, que relacionam os valores das variáveis nodais e os carregamentos aplicados, é obtido através da forma variacional de cada elemento da malha.

Com isso, através de superposição desses sistemas, é possível construir um sistema matricial de equações de todo o problema. Em problemas lineares, o sistema é associado através da matriz de rigidez global (***K***), o vetor de deslocamentos nodais ($\bar{\boldsymbol{u}}$) e do vetor de carregamentos nodais (***F***) como:

$$K * \bar{u} = F$$

Com isso é possível obter um novo sistema de equações envolvendo apenas os valores dos graus de liberdade que eram até o momento desconhecidos. E este sistema é montado através das condições de contorno de Dirichlet ao sistema matricial. Isso resulta na definição completa dos deslocamentos nodais da estrutura estudada.

2.3.2 Matriz de rigidez

É admitido que o sólido apresenta propriedade de isotropia, isto é, as propriedades elásticas são as mesmas, independente da direção. Como consequência da Lei de Hooke, como também o efeito Poisson, a matriz de rigidez do elemento quadrilátero obtida analiticamente e implementada no código de Sigmund

(2000), é uma matriz (8x8) e é expressa na forma matricial de acordo com a seguinte equação:

$$K_e = \frac{E}{1 - v^2} * k$$

onde,

E – Módulo de Young do material sólido

$\boldsymbol{v}$ – Coeficiente de Poisson

$\mathrm{K_e}$ – Matriz de rigidez para um elemento de dimensão unitária.

k – É a matriz auxiliar, que é definida abaixo:

$$k = \begin{pmatrix} k(1) & k(2) & k(3) & k(4) & k(5) & k(6) & k(7) & k(8) \\ k(2) & k(1) & k(8) & k(7) & k(6) & k(5) & k(4) & k(3) \\ k(3) & k(8) & k(1) & k(6) & k(7) & k(4) & k(5) & k(2) \\ k(4) & k(7) & k(6) & k(1) & k(8) & k(3) & k(2) & k(5) \\ k(5) & k(6) & k(7) & k(8) & k(1) & k(2) & k(3) & k(4) \\ k(6) & k(5) & k(4) & k(3) & k(2) & k(1) & k(8) & k(7) \\ k(7) & k(4) & k(5) & k(2) & k(3) & k(8) & k(1) & k(6) \\ k(8) & k(3) & k(2) & k(5) & k(4) & k(7) & k(6) & k(1) \end{pmatrix}$$

Cujo os coeficientes são obtidos através das equações abaixo:

$$k(1) = \frac{1}{2} - \frac{v}{6} \qquad k(5) = -\frac{1}{4} + \frac{v}{12}$$

$$k(2) = \frac{1}{8} + \frac{v}{8} \qquad k(6) = -\frac{1}{8} - \frac{v}{8}$$

$$k(3) = -\frac{1}{4} - \frac{v}{12} \qquad k(7) = \frac{v}{6}$$

$$k(4) = -\frac{1}{8} + \frac{3v}{8} \qquad k(8) = \frac{1}{8} - \frac{3v}{8}$$

A matriz auxiliar k deve ser obrigatoriamente definida positiva (possuem todos os autovalores maiores que zero), tem-se as seguintes propriedades para os coeficientes elásticos E e v:

$$E > 0$$

$$0 < v < \frac{1}{2}$$

Vale salientar que há materiais que o coeficiente de Poisson é negativo podendo chegar a -1, que são os materiais auxéticos, isso significa que eles têm o seu volume aumentado quando aplicado uma força de compressão. Porém, esse tipo de material não é utilizado na engenharia por isso foi considerado como valor mínimo o zero.

Para este trabalho, foram utilizados 1,00 e 0,30 para E e v respectivamente. Pois para obter resultado de densidade dentro do intervalo entre 0 e 1 é necessário que o módulo de elasticidade seja 1. E de acordo com Sigmund (2001), para que a estrutura fosse fabricável é necessário que o coeficiente de Poisson seja 0,3.

2.4 Problemas Numéricos da Otimização Topológica

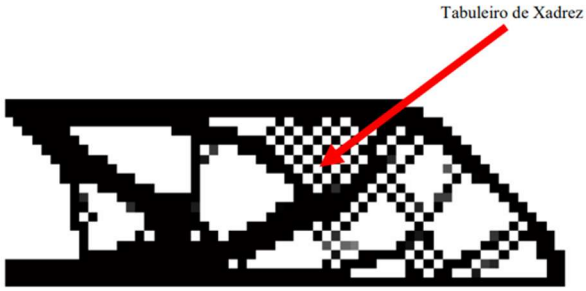
Com a consolidação do processo de OT surgiram linhas de pesquisas e aplicações desses processos. Além dos problemas atrelados ao SIMP, existe o problema de instabilidade numérica. Esses problemas numéricos são pesquisados desde o início do século XXI com os estudos de Sigmund, Haber, Person, entre outros que sugeriram soluções para esses problemas.

2.4.1 Instabilidade de Tabuleiro

O problema numérico de instabilidade de tabuleiro, também conhecido como “checkerboard” na literatura, é caracterizado pela formação na topologia ótima de regiões contendo vazios (ausência de material) e sólidos (presença de material), que se assemelham a um tabuleiro de xadrez como apresentado na Figura 7.

Existem muitos procedimentos e métodos na literatura sobre como tratar esse tipo de problema através de filtros. Existem duas linhas que tem maior aceitação dentro da comunidade sobre as causas desse problema, propostas por Diaz e Sigmund em 1995.

FIGURA 7 – TABULEIRO DE XADREZ PRESENTE DA TOPOLOGIA.



Fonte: Porto, 2006.

A primeira hipótese é que essa instabilidade é devida as aproximações numéricas utilizadas no MEF. Segundo os autores, a configuração em forma de tabuleiro de xadrez traz mais rigidez do que a parte homogênea, quando se compara deformação devido a cisalhamento e com volume constante. Por isso essa configuração aparece como resultado na busca do ótimo. Porém, esse resultado não é coerente com a realidade, a rigidez maior é devido à instabilidade numérica (SIMONETTI, 2009). Uma forma de evitar esse fenômeno é a utilização de filtros, os quais evitam uma mudança muito grande de densidade por meio de avaliação da vizinhança dos elementos, representado neste trabalho pelo raio mínimo (r_{min}).

2.4.2 Dependência de malha

A dependência de malha é o problema de não obter o mesmo resultado para diferentes malhas ou também chamado de discretização do domínio. Quando temos um domínio contínuo e ele é subdividido, o espaço das soluções passa a ter uma dimensão finita, assim a cada discretização mais refinada uma nova solução mais próxima da exata pode ser obtida.

Seria esperado que quanto mais refinada a malha, melhores as estruturas ótimas obtidas. Porém, observa-se que com o aumento da discretização a topologia ótima tende a mudar, aumentando o número de vazios e sua complexidade. Com isso o resultado é qualitativamente diferente de um modelo com uma malha menos refinada.

Vários trabalhos propuseram solucionar esse problema, como o de Bendsoe e Sigmund (2004) e de Sigmund e Peterson (1998), propondo a inclusão de um filtro de independência de malha e diminuir a complexidade da estrutura ótima.

FIGURA 8 – DEPENDÊNCIA DE MALHA COM (A) 3840 ELEMENTOS TRIANGULARES E (B) 19200 ELEMENTOS TRIANGULARES.

Fonte: Simonetti, 2009.

Na Figura 8 é possível verificar o aumento de “ramos” no interior do domínio da (a) para a (b), onde só havia dois “ramos” na figura (a) aumentou para quatro “ramos” e ainda gerou um espaço vazio onde anteriormente era preenchido.

2.5 Manufatura Aditiva (Additive Manufacturing – AM)

Devido à alta complexidade do leiaute das estruturas otimizadas, há a dificuldade de fabricação das peças utilizando métodos tradicionais. Com isso a manufatura aditiva ou também chamada de impressão 3D torna viável a fabricação de tais peças. Além disso, a questão econômica e a exigência cada vez maior em relação à qualidade, resistência e confiabilidade trazem grandes desafios de engenharia em componentes cada vez mais leves e com elevada resistência mecânica.

A manufatura aditiva pode ser definida como um processo de fabricação com base em dados de um modelo tridimensional (3D), por adição de material camada após camada. Pode ser utilizado diferentes tipos de materiais e tecnologias. Independente do material que pode ser utilizado o processo de impressão se inicia com um modelo 3D da geometria almejada, processada em um software de planejamento do processo que fatia a estrutura e determina o caminho que deve ser

impresso. Com isso, o material é depositado camada a camada e a peça e formada sem a necessidade de molde ou fixações.

Zhu, Zhang e Xia (2016) citam que o peso possui grande importância não só nos parâmetros que envolvem a fabricação do produto, mas também seu desempenho e sua vida útil. Peças mais leves requerem, normalmente, menos custo de fabricação e manutenção, além de serem mais sustentáveis. Como a OT visa melhor distribuir o material de um projeto, os dois tópicos se unem para realizar a fabricação de produtos mais leves e mantendo a funcionalidade inicial.

Quanto a aplicação de peças fabricadas pela AM para utilização final, é um grande desafio, tendo em vista que ainda precisam ser exploradas e estudadas a caracterização dos materiais para cada tecnologia e suas propriedades mecânicas. Assim, os projetistas que utilizem o AM necessitam conhecer muito bem o material que está sendo utilizado, consequentemente suas propriedades mecânicas.

Um das grandes dificuldades apontadas por Zhu, Zhang e Xia (2016) é que os profissionais que atuam no desenvolvimento de produtos com AM não foram capacitados para projetar geometrias que são mais complexas, como as que são geradas com o OT. Sendo assim, é de grande importância o aumento recente no estudo de resistência e funcionalidade de componentes otimizados obtidos por AM, buscando entender a complexidade que é relacionar as variáveis de fabricação com os modelos de cálculos estrutural e de otimização.

3. METODOLOGIA

Neste capítulo será descrito o algoritmo utilizado neste trabalho, o código utilizado é o mesmo utilizado por Andreassen e Clausen, *et al.* (2010). Nele conterá uma breve motivação da utilização do MATLAB como ferramenta, análise de elementos finitos, filtro de sensibilidade e o laço de otimização.

3.1 MATLAB

O MATLAB é uma linguagem de programação interativa de alto desempenho voltado para cálculos para as diversas áreas desde equações diferenciais, até estatística, processamento de sinais, finanças e outros. Tem seu próprio *Integrated Development Environment* (IDE) e conjunto de bibliotecas.

O grande destaque do MATLAB é que sua base é uma matriz que não precisa ser dimensionada. Isto é, ela permite o uso de funções que levariam mais tempo em outras linguagens de programação.

Por causa de sua estrutura diferenciada, capacidade de expansão e flexibilidade, o software possui também as ferramentas de elementos finitos, inteligência artificial, depuração de processamento em tempo real e diversas outras soluções.

Também é possível gerar elementos visuais simples, como o que será apresentado neste trabalho, onde é gerado o resultado imediato e depois esse resultado pode ser refinado para a obtenção de uma geometria complexa.

3.2 Lógica do código

Nesta seção será descrito como foi desenvolvido o código de Andreassen e Clausen *et al.* (2010) e como ele pode ser adaptado para as condições de contorno desejadas. Como o código de Andreassen e Clausen *et al.* (2010) tem uma grande

semelhança com o código de 99 linhas de Sigmund (2001), a descrição a seguir também dá parâmetro para o entendimento do código revolucionário de Sigmund.

3.2.1 Condições de Contorno e Análise de Elementos Finitos:

O código empregado é o mesmo utilizado no trabalho de Andreassen e Clausen (2010), um algoritmo de cunho didático e é baseado no código de 99 linhas de Sigmund (2001), que é o mais conhecido código de OT. Os dois modelos são bem parecidos e tem a mesma base de implementação, porém o modelo mais atual possibilita o uso de novos filtros, outros métodos de solução e apresenta uma melhor desempenho no geral.

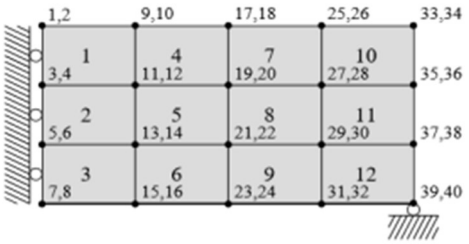
A função de chamada para o algoritmo é: top88(*nelx. nely, volfrac, penal,* r_{min}*, ft).*

onde,

- ***nelx*** e **nely** são os números de elementos na horizontal e vertical, respectivamente da malha desejada;
- **Volfrac** é a fração de volume;
- ***penal*** é o expoente de penalização;
- r_{min} é o filtro para o raio mínimo de avaliação da vizinhança do filtro;
- ***ft*** especifica qual o filtro desejado, se é de sensibilidade (ft = 1) ou de densidade (*ft* = 2).

A geometria utilizada para o domínio são elementos quadrilaterais conforme é representado na Figura 9.

FIGURA 9 – DOMÍNIO DO MODELO COM 12 ELEMENTOS.



Fonte: Andreassen, Clausen, *et al.*, 2010.

É iniciado o pré-processamento do elemento finito definindo as propriedades dos material e determinação de E e $\boldsymbol{v}$. A matriz de rigidez do elemento é replicada para toda a malha, pois os elementos são iguais.

Buscando garantir uma construção eficiente da matriz de rigidez a função $edofMat$ é utilizada. Na matriz as linhas apresentam a sequência de oito graus de liberdade para cada elemento, pois cada nó contém dois graus de liberdade. O armazenamento dos graus de liberada se inicia na extremidade inferior esquerda e continua na direção horária.

A cada iteração do laço de otimização, a construção da matriz global de rigidez **K** é feita pela função $sparce$ que evita armazenar valores vazios sem necessidade. Para garantir que a matriz de rigidez é simétrica é feita a seguinte operação:

$$K' = \frac{(K^T + K)}{2}$$

Se a operação resultar numa matriz simétrica, é possível aplicar a fatoração de Cholesky para resolver as equações de equilíbrio. Caso contrário, o método LU é utilizado (Andreassen, Clausen, *et al*., 2010).

As condições de contorno são inseridas no código pontualmente, onde é necessário identificar o grau de liberdade de interesse, para o correto posicionamento dos apoios e carregamentos. O algoritmo de Andreassen e Clausen tem uma metodologia simples para a alteração das condições de contorno.

3.2.2 Atualização do modelo

Neste trabalho, a atualização é feita de acordo com o trabalho de Andreassen, Clausen, *et al*. (2010), segundo a equação:

$$x_{k+1} = \begin{cases} \max(0, x_k - \alpha) & se \quad x_k * B_k^{\delta} \leq \max(0, x_k - \alpha) \\ \min(1, x_k + \alpha) & se \quad x_k * B_k^{\delta} \geq \min(1, x_k - \alpha) \\ x_k * B_k^{\delta} & qualquer\ outro\ caso \end{cases}$$

x_{k+1}, o novo elemento, depende do passo ($\alpha = 0{,}5$) e da constante δ que é igual a 0,5 e representa o coeficiente de amortecimento. O fator multiplicador B_k é determinado pela equação abaixo:

$$B_k = \frac{-\frac{\partial c}{\partial x_k}}{\lambda \frac{\partial V}{\partial x_k}}$$

onde, $\frac{\partial c}{\partial x_k}$ é a derivada local da flexibilidade e $\frac{\partial V}{\partial x_k}$ é a derivada do volume. λ é o multiplicador de Lagrange e deve ser escolhido para que a restrição de volume seja satisfeita.

A sensibilidade das funções c, que se refere a flexibilidade da estrutura, e do volume V com relação ao elemento k é determinado pelas equações abaixo:

$$\frac{\partial c}{\partial x_k} = -\eta x_k^{\eta-1}(E_0 - E_{min})u^T k u$$

$$\frac{\partial V}{\partial x_k} = 1$$

3.2.3 Filtros: Densidade e Sensibilidade

Os filtros são utilizados para solucionar problemas de instabilidades numéricas, garantir a existência da solução e evitar as formações de tabuleiro de xadrez ou dependência de malha (BOURDIN, 2001).

Os dois filtros presentes no trabalho são os de sensibilidade e de densidade. O primeiro deles atua na derivada da flexibilidade e o segundo atua diretamente na densidade. Segundo Andreassen, Clausen, *et al. (*2010) o filtro de sensibilidade transforma a derivada da densidade conforme a equação abaixo:

$$\widehat{\frac{\partial c}{\partial x_k}} = \frac{1}{\max(y, x_k) * \sum_{i \in N_k} H_{ki}} * \sum_{i \in N_k} H_{ki} * x_i \frac{\partial c}{\partial x_i}$$

onde, y = 10^{-3} é uma constante positiva utilizada para evitar que a divisão seja feita por zero. N_k é o número de elementos próximos ao elemento k. H_{ki} é o fator peso determinado pela equação abaixo:

$$H_{ki} = \max\left(0, r_{min} - \Delta(k,i)\right)$$

em que o r_{min} é o raio mínimo onde o filtro atual, $\Delta(k,i)$ é a distância entre os elementos k e o elemento i. Essa análise de vizinhança com pesagem dos elementos presentes no raio mínimo é feita visando evitar mudanças bruscas na densidade, ocasionando uma instabilidade numérica.

A transformação da densidade original x_k feita pelo filtro de densidades conforme a equação a seguir:

$$\tilde{x}_k = \frac{1}{\sum_{i \in N_k} H_{ki}} \sum_{i \in N_k} H_{ki} * x_i$$

Onde, $\tilde{x}_k$ é a densidade artificial ou também chamada de densidade fictícia. No caso, é aplicado o filtro de densidade as derivadas da flexibilidades e do volume e são determinadas pela regra da cadeia, conforme representado na equação abaixo:

$$\frac{\partial c}{\partial x_j} = \sum_{k \in N_j} \left(\frac{\partial c}{\partial \tilde{x}_k} * \frac{\partial \tilde{x}_k}{\partial x_j} \right) = \sum_{k \in N_j} \left(\frac{1}{\sum_{i \in N_k} H_{ei}} * H_{jk} * \frac{\partial c}{\partial \tilde{x}_k} \right)$$

3.2.4 Laço de otimização

O laço de otimização é iniciado com as variáveis do modelo sendo igualadas a fração do volume definida. Sendo assim, $x_k = \tilde{x}_k$ é verdadeiro caso o filtro de sensibilidade ou de densidades sejam utilizados e as variáveis representam campos homogêneos.

No início da iteração da função objetivo é feita a análise de elementos finitos e depois disso a flexibilidade é computada, assim como as sensibilidades dc e dv da função objetivo e do volume. A função $edofMat$ é utilizada para avaliar a flexibilidade de todos os elementos simultaneamente e a matriz resultante da utilização dessa função é utilizada como índice na matriz de deslocamentos globais U**.** O resultado da utilização da matriz gerada pela função $edofMat$ como índice da matriz de

deslocamento, faz com que seja gerado uma matriz de mesmo tamanho contendo os deslocamentos correspondentes aos graus de liberdade listado no $edofMat$.

Após a função $edofMat$, são aplicados os filtros e finalmente o modelo é atualizado pelo método heurístico. A variável ch é utilizada para verificar a condição de parada do processo de otimização. Ela é iniciada com o valor 0,2 e seus valores seguintes são determinadas pela máxima diferença entre as densidades x_{k+1} e densidade da iteração anterior x_k. A condição de parada é acionada quando essa diferença for menor do que 0,01.

3.3 Descrição geral do código

Nesta sessão será descrito a lógica geral do código desenvolvido por Sigmund (2001), adaptado por Andreassen, Clausen, et al. (2011) e novamente adaptado pelo autor para o desenvolvimento de cada um dos casos propostos.

Para depurar o código é necessário executar MATLAB o seguinte comando $top88(nelx, nely, volfrac, penal, rmin, ft)$. Se for utilizado o filtro de sensibilidade (ft = 1) é possível obter os mesmo resultados que Sigmund (2001) obteve. Porém, quando utilizado o filtro de densidade (ft = 2) normalmente o número de iterações é maior e é possível ter um resultado mais refinado.

A grande diferença entre o código de Sigmund e de Andreassen, *et al*, é que no laço de otimização a função for utilizada para montar as matrizes de elementos finitos, cálculo de conformidade e realizar as operações de filtragem foram vetorizadas. Além disso, os $arrays$ foram construídos por meio de um *loop* que pré-aloca adequadamente, uma boa parte do código está fora do *loop* de otimização, fazendo com que somente o que é necessário seja reprocessado. Todas as sub-rotinas foram colocadas para o início do programa para facilitar o entendimento e implementação de mudanças e foi feito uma distinção das variáveis de projeto e das densidades físicas para facilitar a utilização do filtro de densidade.

O código pode ser resumido em três partes: análise de elementos finitos, filtro e *looping* de otimização. O detalhamento de cada umas das partes se dará abaixo e

a numeração das linhas do código que serão apontadas tem como referência o código original de Andreassen, Clausen, *et al.*, 2010, que pode ser encontrado no Anexo A.

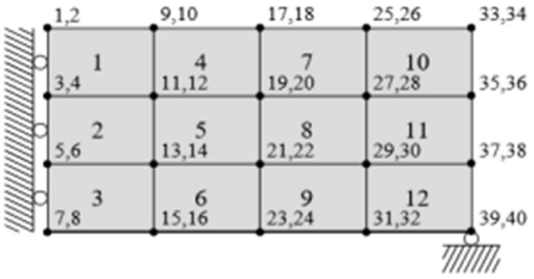
3.3.1 Análise de elementos finitos

Para a análise de elementos finitos, é necessário definir as propriedades do material, isto é feito entre as linhas 4 e 6 do código, onde são colocados os valores de E0 (módulo Young), Emin (módulo de Young artificial), nu (coeficiente de Poisson). Em seguida é calculada a matriz de rigidez k_0 para um elemento de módulo de Young unitário entre as linhas 8 e 12 do código e essa matriz é denotada como KE e por sua regularidade, todos os elementos da malha são idênticos.

Visando permitir uma montagem eficiente da matriz de rigidez no *loop* de otimização a matriz gerada pela função $edofMat$ é construída, entre as linhas 13 e 15. Cada uma das linhas possui 8 colunas e cada valor presente em cada coluna representa a numeração do grau de liberdade (GL) de cada nó. Essa matriz é construída inicialmente achando o número de nós com a expressão $(nelx + 1) * (nely + 1)$, posteriormente a matriz é utilizada para achar o primeiro número do GL de cada nó e é armazenado no local $nelx * nely$ na matriz $edofVec$.

Por fim, a matriz $edofVec$ é remodelada em um vetor coluna e é utilizada para determinar os oito GL de cada elemento. Os resultados dessa remodelação são coletados pela matriz $edofMat$. A Figura 10 é exemplo de geometria utilizado por Andreassen, Clausen, et al. (2010) e mostra como seria gerado a matriz $edofMat$.

FIGURA 10 – DOMÍNIO DO MODELO COM 12 ELEMENTOS.



Fonte: Andreassen, Clausen, *et al*, 2010.

$$\begin{bmatrix} 3 & 4 & 11 & 12 & 9 & 10 & 1 & 2 \\ 5 & 6 & 13 & 14 & 11 & 12 & 3 & 4 \\ 7 & 8 & 15 & 16 & 13 & 14 & 5 & 6 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 31 & 32 & 39 & 40 & 37 & 38 & 29 & 30 \end{bmatrix} \begin{matrix} \leftarrow Elemento\ 01 \\ \leftarrow Elemento\ 02 \\ \leftarrow Elemento\ 03 \\ \vdots \\ \leftarrow Elemento\ 12 \end{matrix}$$

A montagem da matriz K é iniciada na linha 55 do algoritmo por meio da função $sparse$ utilizando vetores (iK e jK) criados nas linhas 16 e 17, que representam os índices de linha e coluna da matriz $edofMat$. As condições de contorno são implementadas da linha 18 até 23 do código e ela é bem parecida com o código de Sigmund, a grande diferença é a retirada desses elementos do *loop* de otimização. Na linha 56 do código o sistema de equações de elementos finitos é resolvido.

3.3.2 Filtro

A aplicação do filtro de sensibilidade envolve uma média ponderada entre diferentes elementos. É uma operação linear, com isso ela pode ser implementada como um produto matricial de uma matriz de coeficientes e um vetor de sensibilidades, está operação é realizada na linha 64 do algoritmo.

A utilização do filtro de densidades implica não só na filtragem de densidades, mas também uma modificação da regra da cadeia das sensibilidades da função objetivo e da restrição de volume. A filtro de densidade é feito na linha 77 do programa e a modificação da sensibilidade é feita entre as linhas 66 e 67 do código em questão.

3.3.3 Looping de Otimização

O $looping$ de otimização é a parte principal do código, ele é iniciado na linha 46. As variáveis de projeto são igualadas, inicialmente, a fração de volume. As densidades físicas também se igualam as variáveis de projeto quando utilizado o filtro de sensibilidade e essa igualdade é sempre válida, enquanto na filtragem de densidades ela é válida quando as variáveis de projeto representam um campo uniforme.

Cada $loop$ otimização começa com a análise de elementos finitos na linha 54 até 56 do programa. Em seguida, a função objetivo (flexibilidade) c é calculada, bem como as sensibilidades dc e dv da função objetivo e da restrição de volume com relação as densidades físicas, esta operação é feita entre as linhas 58 e 61 do código.

As sensibilidades são subsequentemente filtradas (se utilizado o filtro de sensibilidade) ou modificadas (se utilizado o filtro de densidade), nas linhas 63 até 68 do algoritmo.

Entre as linhas 70 e 82 do programa é utilizado um método de otimização para atualizar as variáveis de projeto. A atualização é feita de forma parecida ao código de Sigmund, porém há três diferenças:

I. A sensibilidade dv, restrição de volume, é considerada.

II. O multiplicador de Lagrange é determinado usando as densidades físicas, em vez das variáveis de projeto.

III. A condição de parada é especificada em termos relativos.

Nas linhas 84 e 85 do código é onde os resultados são impressos e são plotados pela linha 87 do código. O critério de parada é dado quando a diferença entre os normas de cada variável de projeto é inferior a 1%.

3.3.4 Resultados

Os resultados são impressos no prompt retornando os valores relativos à quantidade de interações, flexibilidade, a porcentagem do volume restante após a otimização e o resultado da resolução da função objetivo. Além de apresentar o mapa de cores mostrando o design final da estrutura otimizada.

4. RESULTADOS E DISCUSSÕES

Neste capítulo busca realizar a análise dos resultados de cada estudo de caso escolhido. Foram considerados 6 exemplos: viga engastada, viga biapoiada, viga de Michell, viga biengastada, ponte com tabuleiro superior e prédio.

Os problemas analisados foram escolhidos tomando como base trabalhos de Simonetti (2009), Maia (2021) e Nascimento (2019). Com esses exemplos é possível validar o código Andreassen e Clausen *et al.* (2010) com as devidas modificações de condições de contorno. Será avaliado três critérios: a ocorrência de instabilidades, quantidade de iterações necessárias para chegar ao ótimo e diferença de resultados entre os filtros.

É importante informar os parâmetros, que interferem no tempo de processamento, utilizados na máquina para obtenção desses resultados:

- Processador: I7-7700HQ;
- Memória RAM: 16GB – 2.4MHz;
- Placa de vídeo: GTX 1050 TI – 4GB;
- HD: Disco Rígido – 1TB.
- Conectado na Energia
- Sem utilizar internet

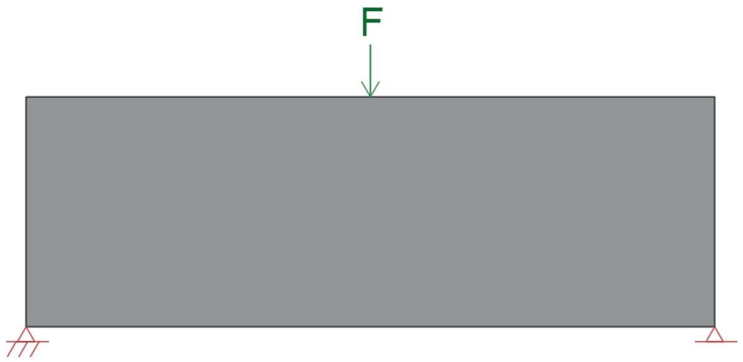
Pois é provável que em máquinas mais recentes sejam obtidos outros resultados com relação ao tempo para o código convergir. Pois houve mudança nos tempos quando a máquina conectou na internet, então há possibilidade de equipamentos mais novos conseguirem convergir o código mais rápido

Vale ressaltar que os resultados obtidos não apresentam unidades, pois o modelo empregado utiliza valores parametrizados para valores unitários ou dentro do intervalo [0,1]. Foi utilizado o valor unitário para a carga (F) e o domínio é proporcional ao tamanho da malha, pois os elementos têm valor unitário (1x1).

4.1 Caso 1 – Viga Biapoiada:

A viga biapoiada é um dos caso mais simples e solucionados da literatura de análise estrutural, com isso é um caso interessante para verificar como seria tal viga com sua forma otimizada. O domínio inicial pode ser visto na Figura 11 e nela é mostrada a estrutura apoiada em suas extremidades inferiores e com uma carga centrada no topo da viga.

FIGURA 11 – DOMÍNIO INICIAL DA ESTRUTURA BIAPOIADA E CARGA CONCENTRADA CENTRADA.

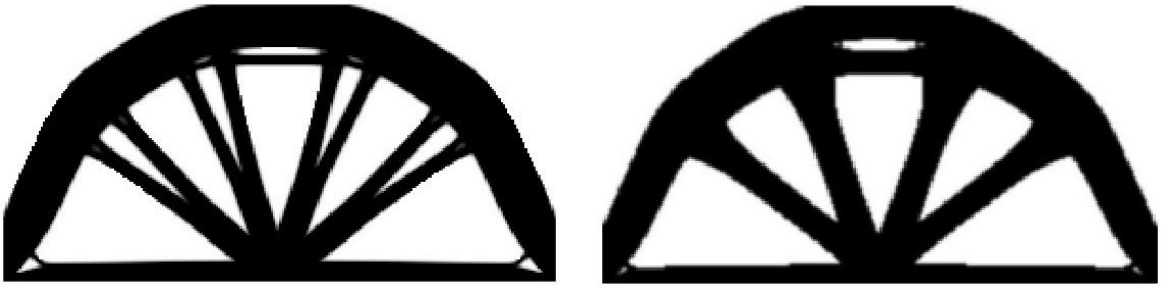


Fonte: Autor, 2023.

O domínio inicial foi discretizado em uma malha de elementos finitos retangulares. Foi tomado como parâmetro o módulo de Young (*E = 1)* com o valor unitário, o coeficiente de Poisson (ν) foi determinado como sendo 0,3 conforme os trabalhos de Sigmund (2001) e Andreassen e Clausen, *et al.* (2010). Foi considerado duas situações para verificação se haveria algum dos problemas numéricos vistos anteriormente, com isso foi considerado uma malha de 500x250 (*nelx* x *nely*) e outra malha de 250x125. Utilizou-se a fração de volume de 50% (volfrac), o raio mínimo (r_{min}) foi de 1,6 e o coeficiente de penalização como 3, pois é um valor que evita valor intermediários dentro do intervalo de 0 e 1.

Para realizar a comparação entre o efeito causado pelos filtros, foi utilizado a mesma malha e modificando o filtro, como pode ser visto na Figura 12. Assim é notado que há diferenças entre as estruturas otimizadas e isso foi devido ao funcionamento dos filtros. Não há presença de zonas cinzas nas estruturas ótimas com filtro, mostrando que os filtros foram eficientes.

FIGURA 12 – ESTRUTURA DE VIGA BIAPOIADA OTIMIZADA UTILIZANDO A MALHA DE 250X125. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.

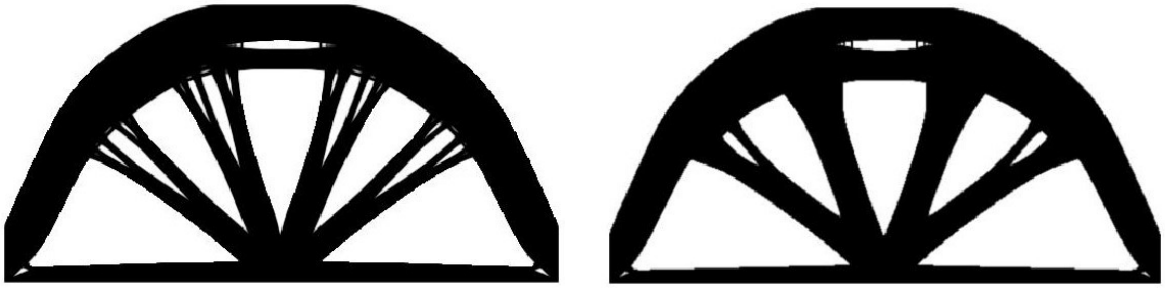


(a) (b)

Fonte: Autor, 2023.

Mas para verificar se há alguma dependência de malha presente, foi otimizada a malha de 500x250, com os mesmos parâmetros de raio mínimo, volume, coeficiente de penalização etc. e o resultado está na Figura 13.

FIGURA 13 – ESTRUTURA DE VIGA BIAPOIADA OTIMIZADA UTILIZANDO A MALHA DE 500X250. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.



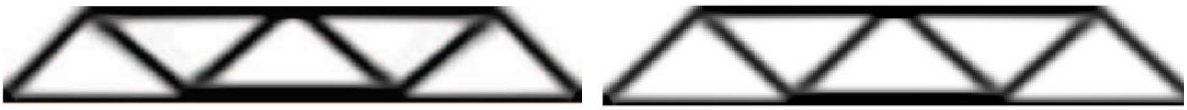
(a) (b)

Fonte: Autor, 2023.

Os filtros apresentaram estruturas diferentes no centro do elemento de estudo, mas não ocorreu a zonas cinzas, mostrando assim que os dois filtros funcionaram para o problema de tabuleiro de xadrez. Porém é possível verificar que há dependência de malha quando utilizado os dois filtros. Isso mostra que o não foi possível evitar o problema numérico, mesmo utilizando os dois filtros.

Como forma de verificação, foi visto os resultados obtidos por Valencia (2012) para a estrutura em questão. Foi verificado que a malha utilizada por Valencia não se assemelhava a estuda neste caso, com isso foi feita uma nova simulação utilizando os parâmetros presente no trabalho de Valencia. A comparação entre os resultados pode ser vista na Figura 14.

FIGURA 14 – ESTRUTURA DE VIGA BIAPOIADA OTIMIZADA UTILIZANDO A MALHA DE 300X50, VOLFRAC = 0,3, RMIN = 10 E O FILTRO DE SENSIBILIDADE. (A) VALENCIA, 2012. (B) AUTOR, 2023.



(a) (b)

Fonte: Autor, 2023.

Isso mostra que o código apresenta resultados semelhantes ao presente em literatura e isso valida o caso.

Como forma de mensurar as diferenças entre a utilização dos filtros, foi extraído do problema o resultado da função objetivo, que é a flexibilidade (c), tanto a inicial quanto a final e o tempo levado para o código convergir.

É mostrado no Quadro 1 a quantidade de iterações necessárias para realizar cada uma das otimizações mostradas neste caso, a variação da função objetivo inicial e final, e o tempo que foi necessário para conseguir convergir o código para obter a estrutura ótima. No quadro abaixo fica claro que com a modificação do filtro altera o resultado da função objetivo.

Quanto mais refinada a malha, maior a necessidade de iterações, maior o tempo para obter a estrutura otimizada e maior o consumo de processamento do computador. Como pode ser notado o filtro de densidade tende a deixar os elementos mais próximos e com isso são mais fáceis de ser fabricado do que os finos ramos apresentados pelas figuras obtidas utilizando o filtro sensibilidade. Mas qualquer uma das configurações propostas consegue resistir as cargas propostas.

QUADRO 1 – RESULTADOS OBTIDO PARA CHEGAR À ESTRUTURA OTIMIZADA DE UMA VIGA BIAPOIADA UTILIZANDO FILTROS E MALHAS DIFERENTES.

Estrutura	***Flexibilidade inicial***	***Flexibilidade final***	***Iterações***	***Tempo para convergência (s)***
Malha 250x125 – Filtro de Densidade	94.3383	16.0185	833	378,012
Malha 250x125 – Filtro de Sensibilidade	94.3383	15.6893	340	170.417
Malha 500x250 – Filtro de Densidade	103.8050	17.0908	1425	2499,186
Malha 500x250 – Filtro de Sensibilidade	103.8050	16.7481	353	652,770

Fonte: Autor, 2023.

4.2 Caso 2 – Viga Engastada Lateralmente:

Este caso representa uma viga em balanço submetida a uma carga concentrada na sua extremidade livre. O domínio inicial pode ser visto na Figura 15.

FIGURA 15 – DOMÍNIO INICIAL DA ESTRUTURA ENGASTADA LATERALMENTE.

Fonte: Autor, 2023.

O domínio inicial foi discretizado em uma malha de elementos finitos retangulares. Foi tomado como parâmetro o módulo de Young (*E = 1)* com o valor unitário, o coeficiente de Poisson (v) foi determinado como sendo 0,3 conforme os trabalhos de Sigmund (2001) e Andreassen e Clausen, *et al.* (2010). Foram considerados os parâmetros de 55% para o volume (*volfrac*), o coeficiente de

penalização de elementos foi tomado como 3 para evitar valores entre 0 e 1, o raio mínimo (r_{min}) estimado foi de 1,60.

Foi escolhido analisar duas malhas para verificar se há presença de problemas numéricos. As malhas escolhidas foram 250x100 e 500x200 e as Figura 16 e Figura 17 apresentam as otimizações de cada uma das malhas.

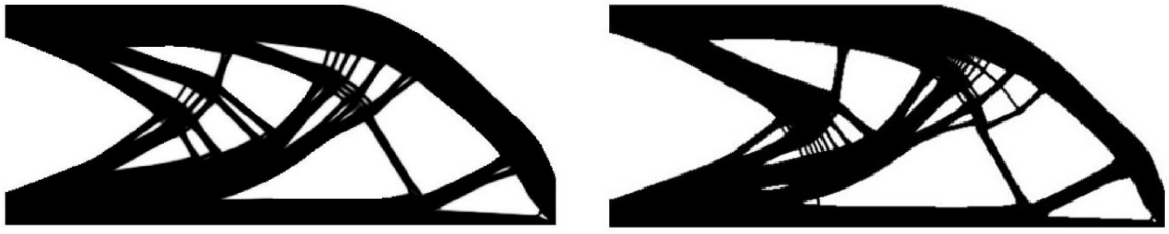
FIGURA 16 – ESTRUTURA OTIMIZADA UTILIZADO A MALHA DE 250X100. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.

(a) (b)

Fonte: Autor, 2023.

A modificação do filtro ocasionou na modificação da estrutura e a mudança foi relativamente grande, já que grande parte das hastes presentes no centro do elemento foram modificadas devido a mudança da forma de filtragem.

FIGURA 17 – ESTRUTURA OTIMIZADA UTILIZADO A MALHA DE 500X200. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.



(a) (b)

Fonte: Autor, 2023.

Utilizando a malha de 500x200 a diferença entre as estruturas otimizadas utilizando o filtro de sensibilidade e o de densidade aumentou. Muito devido a maior

quantidade de elementos a serem otimizados e com isso fornece elementos com uma maior quantidade de detalhes.

Assim, é notado que não há zonas cinzas em nenhuma das quatro imagens, mas há a presença de dependência de malha, pois as estruturas não são iguais e foram modificadas devido a substituição da malha por uma malha com dimensões múltiplas, sendo assim, com as mesmas proporções.

Como no caso 1, as estruturas que utilizaram o filtro de densidade têm ramos mais espessos do que as que utilizaram o filtro de sensibilidade e essa configuração é propícia para o realizar a impressão via manufatura aditiva. Mas ambas as configurações são uteis para suportar as cargas propostas.

No Quadro 2 é demostrado as quantidades necessárias para otimizar cada uma das estruturas demonstradas anteriormente nesse caso e mostra o quão diferente é a quantidade somente modificando o filtro utilizado. Fica claro que a modificação do filtro acaba alterando o resultado da função objetivo.

QUADRO 2 – RESULTADOS OBTIDO PARA CHEGAR À ESTRUTURA OTIMIZADA DE UMA VIGA ENGASTADA UTILIZANDO FILTROS E MALHAS DIFERENTES.

Estrutura	***Flexibilidade inicial***	***Flexibilidade final***	***Iterações***	***Tempo para convergência (s)***
Malha 250x100 – Filtro de Densidade	478.4480	110.6383	829	405.606
Malha 250x100 – Filtro de Sensibilidade	478.4480	107.9186	275	128.115
Malha 500x200 – Filtro de Densidade	487.4047	111.2689	1413	2237.429
Malha 500x200 – Filtro de Sensibilidade	487.4047	108.7606	376	476,281

Fonte: Autor, 2023.

Para validar que o código obteve resultados similares ao presente em literatura, foi analisado o trabalho de Nascimento (2019), onde tem a mesma estrutura. Porém devido a relação entre a altura e comprimento (h/l), e a diferença entre as malhas, as

imagens não são iguais as presente no estudo. Mas utilizando os mesmo parâmetros, a mesma relação entre h/l e comparando os resultados presente na Figura 18. Com isso é possível verificar que as estruturas são similares.

FIGURA 18 – ESTRUTURA OTIMIZADA UTILIZADO A MALHA DE 320X200, RMIN=12, VOLFRAC = 0,4 E FILTRO DE SENSIBILIDADE. (A) NASCIMENTO, 2019. (B) AUTOR, 2023.

(a) (b)

Fonte: Autor, 2023.

Visando provar que os resultados obtidos no MATLAB podem ser fabricados, foi considerado a estrutura otimizada com o filtro de sensibilidade e malha de 500x200. Foi utilizado o software *Inkscape* para vetorizar a imagem obtida pelo MATLAB e o programa *Thinkercard* para modelar a estrutura. Além disso, foi utilizada a impressora *Faber S - Pcyes Flashforge* para realizar a manufatura aditiva da peça em questão, que pode ser vista na Figura 19. O filamento utilizado para a impressão foi o PLA (Poliácido Lático), que é um material biodegradável feito de amido de milho.

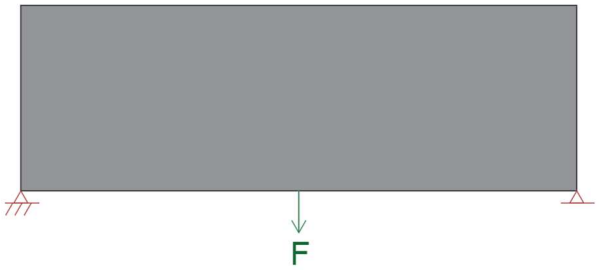
FIGURA 19 – IMPRESSÃO VIA MANUFATURA ADITIVA DE ESTRUTURA OTIMIZADA DE VIGA ENGASTADA E COM UMA CARGA CONCENTRADANA EXTREMIDADE LIVRE.

Fonte: Autor, 2023.

4.3 Caso 3 – Viga de Michel:

A estrutura considerada neste exemplo foi estudada por Michell em 1904 e na figura abaixo mostra como seria a solução analítica dessa estrutura. É um problema comum para algoritmos de OT. O domínio consiste em um retângulo com as extremidades apoiadas e submetido uma carga concentrada no meio do vão ou distribuída ao longo de toda a face inferior da viga, neste problema foi considerado uma carga concentrada. O código inicia com o domínio demonstrado na Figura 20 e com o desenvolvimento do algoritmo é possível chegar à estrutura ótima.

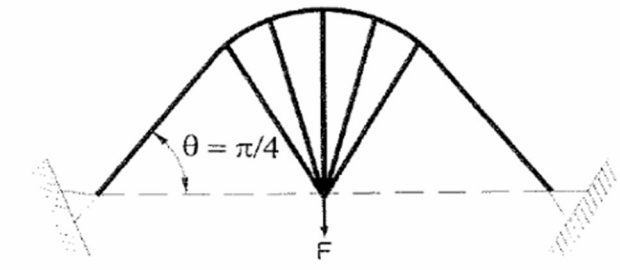
FIGURA 20 – DOMÍNIO INICIAL DA ESTRUTURA DE MICHELL.



Fonte: Autor, 2023.

Na Figura 21 é possível verificar a solução analítica do problema que foi demonstrada por Simonetti (2009).

FIGURA 21 – SOLUÇÃO ANALÍTICA DA ESTRUTURA DE MICHELL.

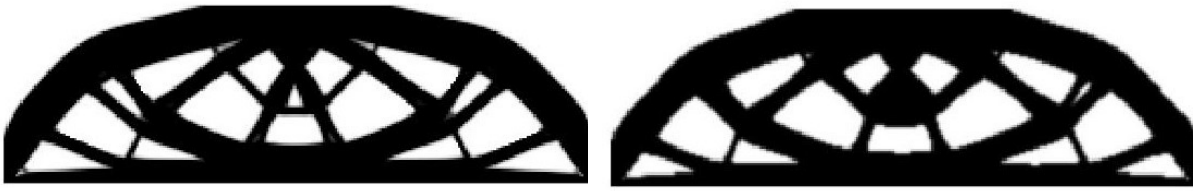


Fonte: Simonetti (2009).

O domínio inicial foi discretizado numa malha de elementos finitos retangulares. Foi considerado o coeficiente de Poisson como 0,3 (ν) e módulo de Young (E) como sendo 1. Foi considerado como parâmetros de solução o volume de 60% (*volfrac)* do volume inicial, o expoente de penalização foi 3 (*penal)*, pois segundo Sigmund (2001)

proporciona resultados “realizáveis” e raio mínimo de 1,5 (r_{min}). Utilizando a malha de 250x75 foram otimizadas duas estruturas com filtros diferentes, conforme mostrado na Figura 22.

FIGURA 22 – ESTRUTURA DE MICHELL OTIMIZADA UTILIZANDO A MALHA DE 250X75. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.

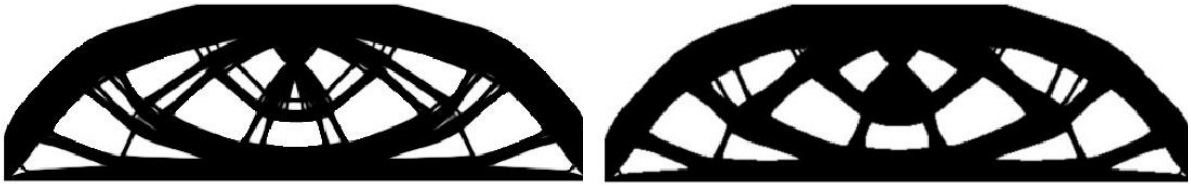


(a) (b)

Fonte: Autor, 2023.

Agora utilizando a malha de 500x150 foi realizado o mesmo experimento de utilizar os filtros de sensibilidade e densidade para verificar a ocorrência de estruturas diferentes devido a utilização de filtros diferente.

FIGURA 23 – ESTRUTURA DE MICHELL OTIMIZADA UTILIZANDO A MALHA DE 500X150. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE



(a) (b)

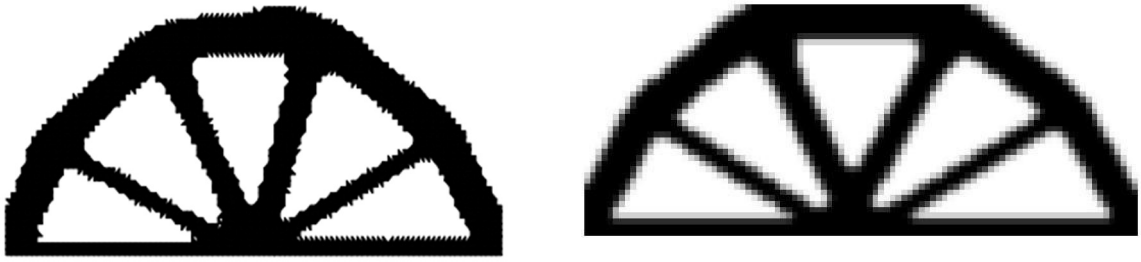
Fonte: Autor, 2023.

Com isso, foi possível verificar a ocorrência do fenômeno de dependência de malha e ocorreu uma leve mudança, mas mantendo sua macroestrutura independente do filtro utilizado e da malha que foi utilizada.

Para verificar se os resultados são similares aos presentes na literatura, foi comparado o trabalho desenvolvido por Simonetti (2009) sobre a viga de Michel. O domínio estudado por eles é diferente do estudado neste caso e a relação entre altura

e comprimento é determinante para a configuração da estrutura. Com isso foi considerado os mesmo parâmetros utilizados no trabalho de Simonetti (2009), mesmo eles utilizando o método SESO. Na Figura 24 pode ser visto a comparação entre os resultados obtidos por Simonetti e o do código presente neste trabalho.

FIGURA 24 – ESTRUTURA DE MICHELL OTIMIZADA UTILIZANDO A MALHA DE 96X40 UTILIZANDO RMIN = 2,0, VOLFRAC = 0,44, FILTRO DE DENSIDADE. (A) SIMONETTI, 2009. (B) AUTOR, 2023.



(a) (b)

Fonte: Autor, 2023.

As estruturas não são idênticas devido a diferença entre os métodos e a geometria dos elementos finitos, pois no trabalho de Simonetti é utilizado elementos finitos triangulares e neste trabalho é utilizando elementos finitos retangulares. Mas as estruturas são semelhantes e pode ser considerado que o código obteve resultados condizentes com o presente na literatura.

Os filtros apresentam uma estrutura semelhante, como pode ser visto nas Figura 22 e Figura 23, mas as imagens apresentadas na Figura 22 apresenta um melhor modelo para ser fabricado, devido a menor presença de estruturas finas. Há uma grande diferença no número de iterações necessárias para chegar na estrutura ótima, conforme é mostrado no Quadro 3. Pode ser visto também que a modificação do filtro alterou o resultado da função objetivo.

Nenhum dos dois filtros foi eficiente em evitar o problema de dependência de malhas, mas evitou que aparecessem zonas cinzas em todas as estruturas. Mesmo com a quantidade de tempo maior para convergir os códigos utilizando o filtro de densidade, ele apresentou estruturas melhores para a fabricação.

QUADRO 3 – RESULTADOS OBTIDO PARA CHEGAR À ESTRUTURA OTIMIZADA DE UMA VIGA MICHELL UTILIZANDO FILTROS E MALHAS DIFERENTES.

Estrutura	***Flexibilidade inicial***	***Flexibilidade inicial***	***Iterações***	***Tempo para convergência (s)***
Malha 250x75 – Filtro de Densidade	82,0201	23,9963	529	185,602
Malha 250x75 – Filtro de Sensibilidade	82,0201	23.6524	280	112,849
Malha 500x150 – Filtro de Densidade	86,3713	24,5790	1404	1696,140
Malha 500x150 – Filtro de Sensibilidade	86,3713	24,2056	380	440,964

Fonte: Autor, 2023.

Visando provar que o resultado obtido no MATLAB pode ser fabricado, foi considerado a estrutura otimizada com o filtro de densidade e malha de 500x150. Foi utilizado o software *Inkscape* para vetorizar a imagem obtida pelo MATLAB e o programa *Thinkercard* para modelar a estrutura. Além disso, foi utilizada a impressora Creality CR-200b para realizar a manufatura aditiva da peça em questão, que pode ser vista na Figura 25.

Para a impressão, foi considerado uma espessura de 3cm, o comprimento de 14,4cm e altura de 4,3cm. O filamento utilizado para a impressão foi o PLA (Poliácido Lático), que é um material biodegradável feito de amido de milho.

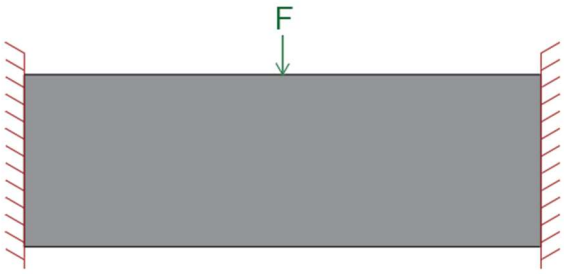
FIGURA 25 – ESTRUTURA OTIMIZADA DA VIGA DE MICHELL IMPRESSA UTILIZANDO MANUFATURA ADITIVA.

Fonte: Autor, 2023.

4.4 Caso 4 – Viga Biengastada:

A viga biengastada é um caso comum nas construções, onde uma viga se apoia em outras duas vigas ou uma viga que se apoia em dois pilares passantes. A Figura 26 apresenta a o domínio da estrutura.

FIGURA 26 – DOMÍNIO INICIAL DA VIGA BIENGASTADA.

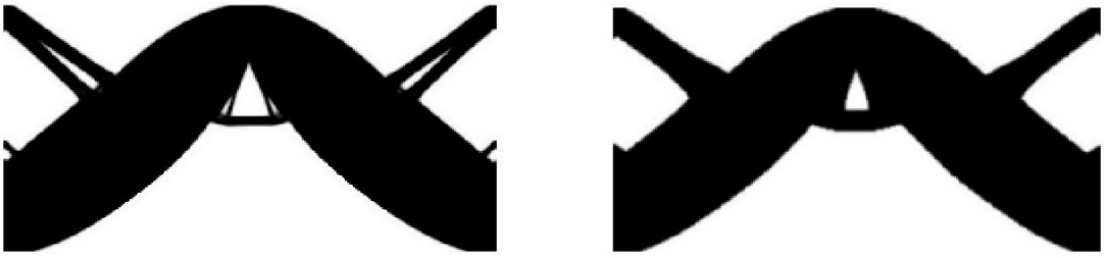


Fonte: Autor, 2023.

O domínio inicial foi discretizado numa malha de elementos finitos retangulares. Foi considerado o coeficiente de Poisson como 0,3 (*η)* e módulo de Young como sendo 1. Foi considerado como parâmetros de solução o volume de 55% (*volfrac)* do volume inicial, o expoente de penalização foi 3 (*penal)*, pois segundo Sigmund (2001) proporciona resultados “realizáveis” e raio mínimo de 1,8 (r_{min}).

Utilizando a malha de 300x150 foi realizado a comparação visual da otimização utilizando o filtro de densidade (*ft = 2*) e utilizando o filtro de sensibilidade (*ft = 1*) e com isso foram obtidas as imagens apresentadas na Figura 27.

FIGURA 27 – ESTRUTURA OTIMIZADA DA VIGA BIENGASTADA UTILIZANDO A MALHA DE 300X150. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE

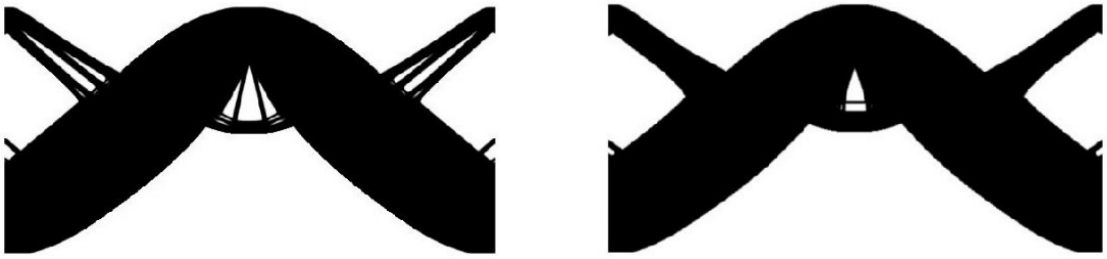


(a) (b)

Fonte: Autor, 2023.

Utilizando uma nova malha, para verificar se há algum problema numérico, foi considerada a malha 600x300, utilizando os filtros de densidade e sensibilidade foi possível obter as estruturas apresentadas na Figura 28.

FIGURA 28 – ESTRUTURA OTIMIZADA DA VIGA BIENGASTADA UTILIZANDO A MALHA DE 600X300. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.



(a) (b)

Fonte: Autor, 2023.

Tal estrutura é semelhante a obtida por Simonetti (2009) que utilizou o outro MOT e obteve a estrutura conforme é mostrado na Figura 29.

FIGURA 29 – ESTRUTURA DE VIGA BIENGASTADA OTIMIZADA UTILIZANDO O MÉTODO SESO COM MALHA DE 60X30 E COM ELEMENTOS FINITOS TRIANGULARES.

Fonte: Simonetti, 2009.

Para verificar se a relação entre a altura e o comprimento influenciava na geometria otimizada, foi simulado a malha de 600x200 e com isso foi obtida os resultado apresentados na Figura 30.

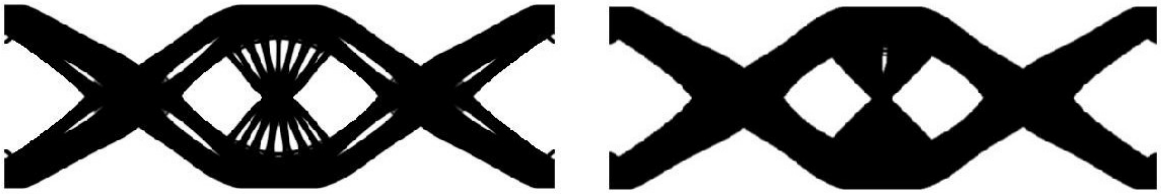
FIGURA 30 – ESTRUTURA OTIMIZADA DA VIGA BIENGASTADA UTILIZANDO A MALHA DE 600X200. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.

(a) (b)

Fonte: Autor, 2023.

Para verificar se o filtro evitou o problema numérico de dependência de malha, foi utilizado a malha de 300x100, utilizando os mesmo parâmetros e os resultados são mostrados na Figura 31.

FIGURA 31 – ESTRUTURA OTIMIZADA DA VIGA BIENGASTADA UTILIZANDO A MALHA DE 300X100. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.



(a) (b)

Fonte: Autor, 2023.

Conforme mostrado nas imagens anteriores, não há a presença de zonas cinzas em nenhuma das imagens. Além disso, é plausível afirmar que das quatro estruturas estudadas nesse caso a única que não teve o problema de dependência de malha foi quando utilizado a malha de 600x300 com as suas proporções quando empregado o filtro de densidade. Todas as outras propostas caíram no problema de dependência de malha.

É possível verificar também a o quão importante é a relação entre a altura e o comprimento para a geometria da estrutura, pois com uma modificação simples modificou drasticamente a geometria final da peça. Então o domínio inicial é fundamental para a geometria otimizada.

Como mostrado no Quadro 4, a quantidade de iterações necessárias para obter a estrutura otimizada é muito diferente devido a aplicação do filtro e do aumento de elementos presentes na malha. Sendo assim, é constatado que o filtro de densidade demanda maior quantidade de iterações do que o filtro de sensibilidade. Além disso, é possível verificar que a modificação do filtro tem influência direta no resultado da função objetivo, pois a flexibilidade final não é a mesma para os casos semelhantes.

QUADRO 4 – RESULTADOS OBTIDO PARA CHEGAR À ESTRUTURA OTIMIZADA DE UMA VIGA BIENGASTADA UTILIZANDO FILTROS E MALHAS DIFERENTES.

Estrutura	***Flexibilidade inicial***	***Flexibilidade final***	***Iterações***	***Tempo para convergência (s)***
Malha 300x150 – Filtro de Densidade	31,8438	6,6571	577	426,164
Malha 300x150 – Filtro de Sensibilidade	31,8438	6.6285	221	154,987
Malha 300x100 – Filtro de Densidade	31,2106	8,5721	628	339,643
Malha 300x100 – Filtro de Sensibilidade	31,2106	8,5183	139	65,574
Malha 600x300 – Filtro de Densidade	34,4974	7,0688	1230	4053,457
Malha 600x300 – Filtro de Sensibilidade	34,4974	7.0375	342	1024,541
Malha 600x200 – Filtro de Densidade	32,3822	8,6409	1242	3024,376
Malha 600x200 – Filtro de Sensibilidade	32,3822	8,5556	326	614,720

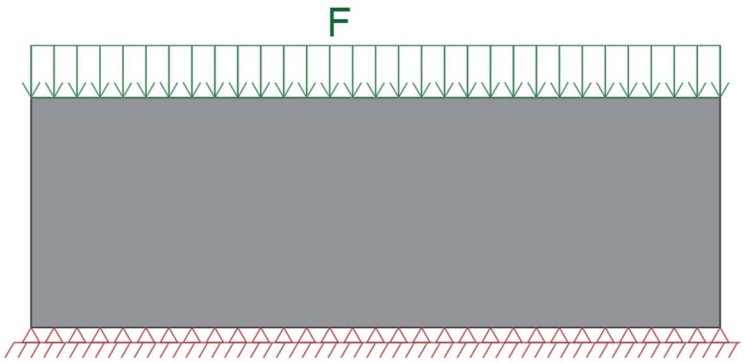
Fonte: Autor, 2023.

4.5 Caso 5 – Ponte com Tabuleiro Superior:

Neste caso será estudado a configuração de uma ponte com o tabuleiro na sua parte superior e com carga distribuída em seu tabuleiro. O nome deste caso foi dado devido a configuração apresentada nos resultados e não pela configuração do domínio inicial. A ponte é um elemento muito comum na construção civil e analisar um

caso como este é importante, pois é possível verificar se a estrutura otimizada se aproxima com as configurações gráficas de uma ponte comum. O domínio inicial é descrito pela Figura 32.

FIGURA 32 – DOMÍNIO INICIAL DA ESTRUTURA DE PONTE COM TABULEIRO NA PARTE SUPERIOR.



Fonte: Autor, 2023.

Para a otimização, a estrutura foi discretizada em duas malhas 600x200 e 300x100 de elementos retangulares. Foi considerado o coeficiente de Poisson como 0,3 (*η)* e módulo de Young como sendo 1. Foi considerado como parâmetros de solução o volume de 45% (*volfrac)* do volume inicial, o expoente de penalização foi 3 (*penal)* e raio mínimo de 1,7 (r_{min}).

FIGURA 33 – ESTRUTURA OTIMIZADA DE UMA PONTE COM TABULEIRO SUPERIOR UTILIZANDO A MALHA DE 300X100. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.

(a) (b)

Fonte: Autor, 2023.

A otimização da malha 300x100 (Figura 33), onde foram utilizados filtros de sensibilidade e densidade foi obtido estruturas graficamente diferentes, pois como nos

outros casos o filtro de sensibilidade fornece estruturas com o material mais espaçado do que o filtro de densidade que busca deixar o material mais próximo. Mas a quantidade de “pilares” presente nas imagens são iguais e a configuração de uma forma geral é bem parecida.

FIGURA 34 – ESTRUTURA OTIMIZADA DE UMA PONTE COM TABULEIRO SUPERIOR UTILIZANDO A MALHA DE 600X200. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.

(a) (b)

Fonte: Autor, 2023.

Utilizando a malha de 600x200, que é o dobro da malha inicial, foram alcançados os resultados presentes na Figura 34. É possível verificar que não houve o problema numérico de tabuleiro de xadrez em nenhuma das quatro imagens. Com relação ao problema de dependência de malha, é visível que com a utilização do filtro de sensibilidade o problema não foi resolvido, mas com a utilização do filtro de densidade é possível ver que há diferenças pequenas entre as duas imagens e isso pode ser considerado que não há dependência de malha.

Como um exemplo de uma estrutura parecida com a obtida nas Figura 33 e Figura 34 temos a estrutura do Centro de convenções do Qatar, que já foi mostrado na Figura 1. As estruturas se assemelham, por apresentar um comportamento como o da raiz das árvores sustentam todo o peso.

A quantidade de iterações necessárias para otimizar cada uma das estruturas mudam bastante. O que mais impressiona é que a quantidade de iterações necessárias para otimizar a estrutura com malha de 300x100 e filtro de sensibilidade foi muito próxima à da estrutura coma malha de 600x200 com filtro de sensibilidade, porém o tempo para convergir o código é quase o dobro, conforme pode ser visto no Quadro 5. Além disso, é possível verificar que a modificação do filtro tem influência direta no resultado da função objetivo, pois a flexibilidade final não é a mesma.

QUADRO 5 – RESULTADOS OBTIDOS PARA CHEGAR À ESTRUTURA OTIMIZADA DE UMA PONTE DE TABULEIRO SUPERIOR UTILIZANDO FILTROS E MALHAS DIFERENTES.

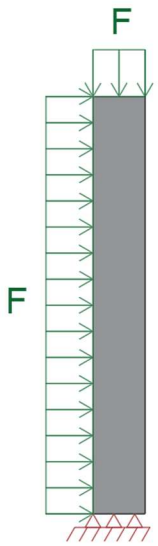
Estrutura	***Flexibilidade inicial***	***Flexibilidade final***	***Iterações***	***Tempo para convergência (s)***
Malha 300x100 – Filtro de Densidade	320.067,6488	86.686,5542	971	1320,144
Malha 300x100 – Filtro de Sensibilidade	320.067,6488	82.154,1264	298	430,992
Malha 600x200 – Filtro de Densidade	1.275.148,3075	344.502,6271	1471	4006,731
Malha 600x200 – Filtro de Sensibilidade	1.275.148,3075	331.735,5567	313	821,954

Fonte: Autor, 2023.

4.6 Caso 6 – Prédio:

Este caso se trata da simulação da estrutura de um prédio otimizado topologicamente, quando submetida a cargas horizontais e verticais. Já há algumas construções onde a fachada do prédio foi feita de acordo com OT e por isso é uma caso interessante de ser estudado. Na Figura 35 pode ser visto as condições de contorno utilizadas no caso.

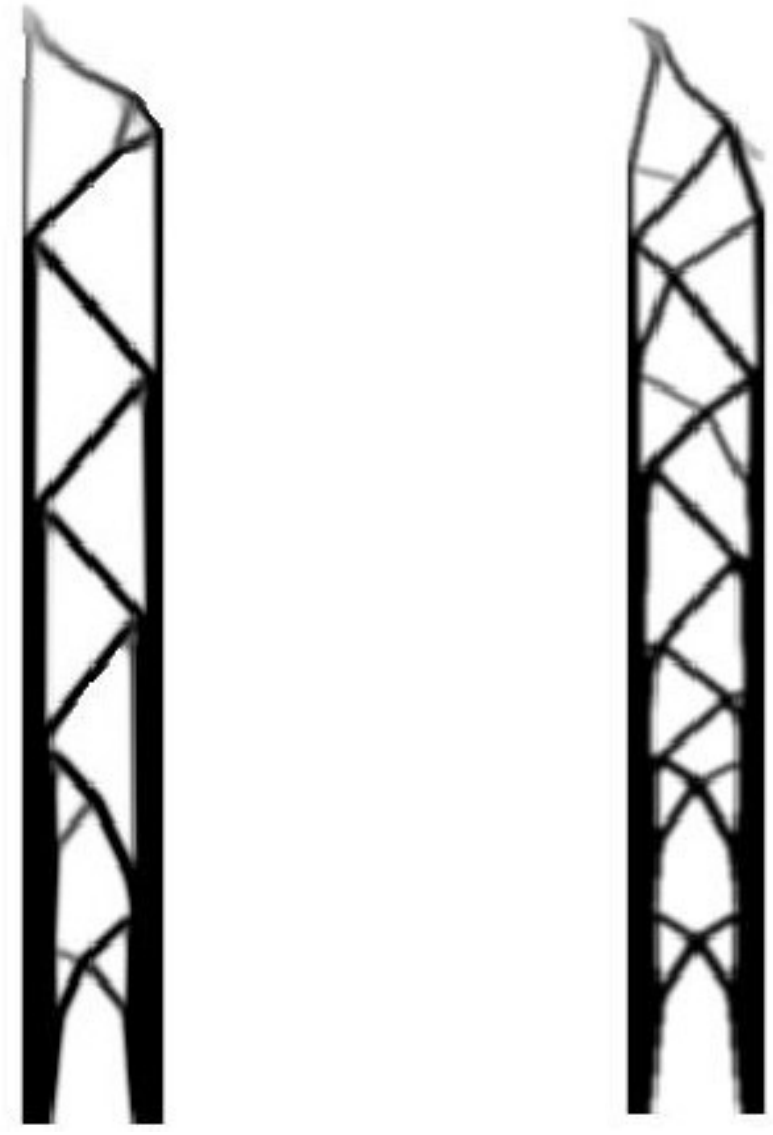
FIGURA 35 – DOMÍNIO INICIAL DE UM PRÉDIO.



Fonte: Autor, 2023.

Para a otimização, a estrutura foi discretizada em duas malhas 800x100 e 400x50 de elementos retangulares. Foi considerado o coeficiente de Poisson como 0,3 (*η)* e módulo de Young como sendo 1. Foi considerado como parâmetros de solução o volume de 40% (*volfrac)* do volume inicial, o expoente de penalização foi 3 (*penal)* e raio mínimo de 3,0 (r_{min}).

FIGURA 36 – ESTRUTURA OTIMIZADA DE UM PRÉDIO UTILIZANDO A MALHA DE 400X50. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.

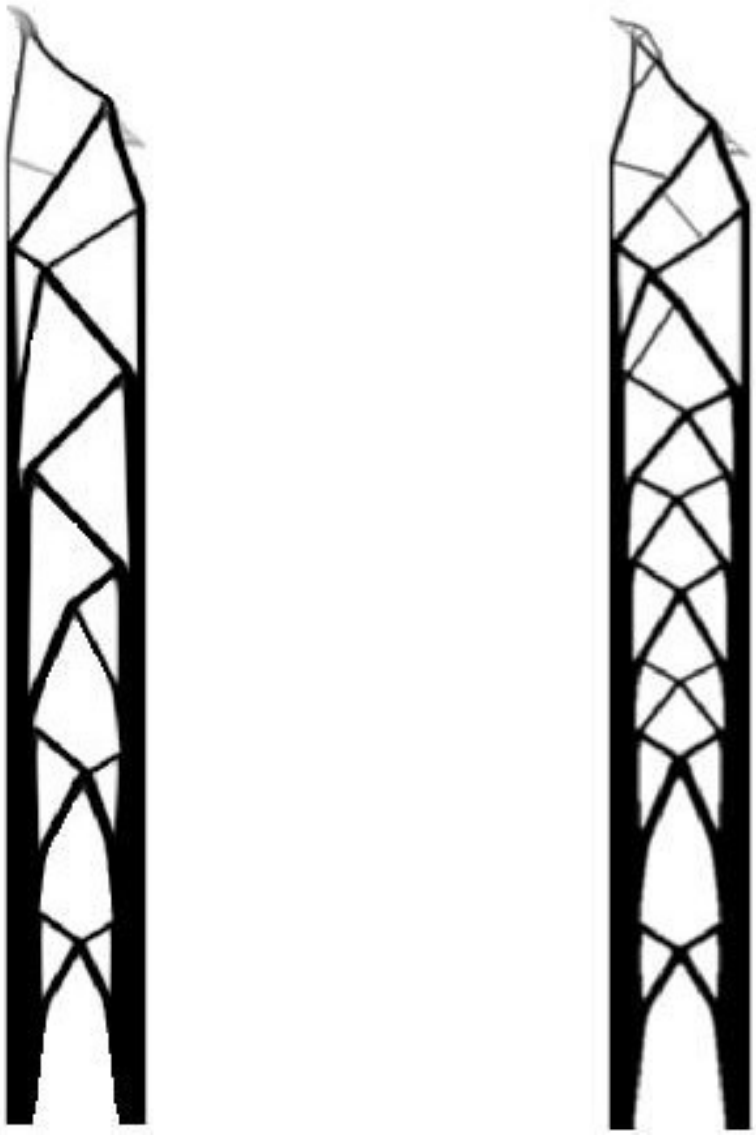


(a) (b)

Fonte: Autor, 2023.

Foi utilizado a malha de 800x100 para verificar a ocorrência de algum problema numérico, onde foram mantidos todos os parâmetros apresentados anteriormente e os resultados podem ser vistos na Figura 37.

FIGURA 37 – ESTRUTURA OTIMIZADA DE UM PRÉDIO UTILIZANDO A MALHA DE 800X100. (A) FILTRO DE SENSIBILIDADE (B) FILTRO DE DENSIDADE.



(a) (b)

Fonte: Autor, 2023.

Como pode ser visto, houve o problema numérico de dependência de malha, pois houve modificação nas estruturas quando modificado a malha, além disso houve

também a presença de zonas cinzas em todos os resultado, mostrando assim que o filtro não foi eficaz em combater tais problemas.

No Quadro 6, é possível verificar o tempo que foi necessário para convergir o código, a quantidade de iterações necessárias e o quanto modificou a flexibilidade para chegar ao ótimo. É notado que os filtros alteram os valores das funções objetivos.

Também é fácil de notar a distribuição do material ao longo das estruturas, onde na porção inferior há mais material e conforme aumenta a altura a quantidade de material é menor.

QUADRO 6 – RESULTADOS OBTIDOS PARA CHEGAR À ESTRUTURA OTIMIZADA DE UM PRÉDIO UTILIZANDO FILTROS E MALHAS DIFERENTES.

Estrutura	***Flexibilidade inicial***	***Flexibilidade final***	***Iterações***	***Tempo para convergência (s)***
Malha 400x50 – Filtro de Densidade	172.156.155,493	6.867.856,4609	1354	1729,980
Malha 400x50 – Filtro de Sensibilidade	172.156.155,493	6.505.734,8068	397	499,032
Malha 800x100 – Filtro de Densidade	271.012.901,7145	24.960.607,2589	1833	4016,958
Malha 800x100 – Filtro de Sensibilidade	271.012.901,7145	24.035.921,5374	933	1922,221

Fonte: Autor, 2023.

Como em todos os outros casos, a quantidade de iterações para se chegar à estrutura ótima com o filtro de densidade é muito maior do que utilizando o filtro de sensibilidade, porém o filtro de densidade fornece soluções mais simples de se aplicar a manufatura aditiva.

Na literatura, temos exemplos similares aos resultados encontrados, pois no estudo de Pereira (2018) sobre o sistema de contraventamento de prédios utilizando otimização topológica, como pode ser visto na Figura 38. O estudo tem carregamentos

diferentes do caso em questão, mas já é possível visualizar que há uma semelhança entre as estruturas. Com isso, mostra que o código forneceu resultados confiáveis.

FIGURA 38 - SISTEMA DE CONTRAVENTAMENTO UTILIZANDO O MÉTODO SIMP E COM CARGA EM TODOS OS PAVIMENTOS.

Fonte: Pereira, 2018.

Os casos propostos são exemplos utilizados cotidianamente na engenharia civil e por isso foram escolhidos, além de muitos deles serem casos já estudados na literatura e com isso foram tomados como parâmetros para verificar se o código está funcionando da mesma forma que apresentado por outros autores.

O código de Andreassen e Clausen (Anexo A) é um dos código mais conhecidos no meio do estudo da otimização topológica, pois é um dos código mais simples entendimento. Para cada um dos casos propostos anteriormente o código deles foi tomado como base e foi modificado a condição de contorno para que fosse adequado para o problema. A lista dos códigos modificados é apresentada nos seguintes anexos:

- No Anexo B pode ser visto a adaptação feita nas condições de contorno para a viga biapoiada com a carga centrada no seu topo.

- O código para o caso da viga em balanço pode ser visto no Anexo C, é um caso parecido com o exemplo de Andreassen e Clausen, mas foi modificado a localização da carga.

- O Anexo D, pode ser visto o código para o caso da viga de Michell. Sendo esse caso proposto em 1904 e muito estudado, por isso foi uma excelente escolha para verificar se o código está retornando um resultado coerente.

- Pode ser visto no Anexo E o código da adaptado para a viga biengastada, este problema também é um dos casos bem desenvolvidos no meio acadêmico e como pode ser visto, os resultados achados foram similares aos presentes na literatura.

- O caso 5 é um problema de uma ponte com seu tabuleiro na parte superior da estrutura e a adequação feita para as condições de contorno do problema pode ser vista no Anexo E. Nele é possível verificar que a estrutura otimizada é similar as raízes de uma árvore sustentando o peso.

- O problema do prédio, caso 6, foi uma sugestão do Professor Renato Motta e é tema do estudo de mestrado de Rayanne Pereira. Com isso se mostrava um problema interessante de ser solucionado e com as considerações de carga e restrições foi possível obter um resultado próximo ao que Rayanne havia obtido no seu estudo de sistemas de contraventamentos para edifícios. O código utilizado por ser visto no Anexo F.

5. CONCLUSÕES

Este trabalho tem o objetivo principal do desenvolver um raciocínio teórico experimental do método de otimização topológica utilizando o SIMP, buscando redução de volume do material sem comprometer a resistência da peça em estudo. Como detalhado ao longo do texto, foi exposto como pode ser desenvolvido o raciocínio para obtenção de uma estrutura topologicamente otimizada.

Outro objetivo foi expor como é o funcionamento de um código de otimização topológica e foi escolhido o código de Andreassen, Clausen, et al. de 2009 (Anexo A) que é mais conhecido como top88 por só ter 88 linhas executáveis de código, este código foi escolhido por ser um dos códigos recentes mais simples e que apresentam bons resultados. Os detalhes do desenvolvimento do código são expostos ao longo do trabalho.

Durante a implementação computacional foi observado a presença de instabilidades numéricas nos resultados. O filtros de sensibilidade, proposto por Sigmund, cujo objetivo é evitar zonas cinzas nas estruturas resultantes, obteve bons resultados e conseguiu cumprir seu papel. Quando ao filtro de densidade, que além de evitar a presença do fenômeno de tabuleiro de xadrez deveria evitar o fenômeno da dependência de malha, porém só conseguiu obter êxito em alguns dos casos.

Também com relação aos filtros, é demonstrado em cada um dos casos que mantendo a malha e modificando o filtro há interferência direta na configuração da estrutura e em todos os casos a mudança de suaves, pois o filtro de densidades fez com que o material ficasse mais junto do que o filtro de sensibilidade e isso faz com que as estruturas que utilizaram o filtro de densidade sejam mais fáceis de serem fabricadas, pois a complexidade de imprimir as estruturas com estruturas mais finas o risco de ocorrência erro na impressão é grande. Com isso, o filtro de densidade se mostra mais produtivo do que o filtro de sensibilidade.

O filtro de densidade apresenta uma quantidade muito maior de iterações para chegar à convergência do que o filtro de sensibilidade. Consequentemente, o tempo para obter a estrutura ótima com o filtro de densidade é maior.

No caso da viga biengastada, ocorreu um fenômeno bem diferente, pois com uma leve modificação na malha a estrutura topológica alterou drasticamente, isso devido a mudança única e exclusivamente devido a proporção entre a altura e comprimento da malha. O resultado de um dos casos é similar ao obtido por Simonetti (2009) utilizando o método SESO (otimização estrutural evolucionária suavizada).

É evidente que quanto maior a malha empregada, maior será a quantidade de detalhes da estrutura e maior será sua nitidez. Em todos os casos as estruturas ótimas mais nítidas foram as que tinham as maiores malhas.

Por fim, como sugere-se para trabalhos futuros utilizar os códigos que desenvolvem estruturas tridimensionais, empregar novos filtros, modificar a função objetivo, considerar incertezas no carregamento, verificar a relação entre problemas numéricos e o raio mínimo de atuação, e testar novos métodos de otimização topológica.

REFERÊNCIAS

AKIN, J. E. **Finite Element Analysis with Error Estimators. An Introduction to the FEM and Adaptive Error Analysis for Engineering Students**. Oxford. 2005.

ANDREASSEN, E.; CLAUSEN, A.; SCHEVENELS, M.; LAZAROV, B. S.; SIGMUND O. **Efficient topology optimization in MATLAB using 88 lines of code. Structural and Multidisciplinary Optimization.** v. 43, p. 1-16, 2010. ISSN 1.

ARORA, J. S. **Introduction to Optimum Design**. 3. ed. Amsterdam: Elsevier Academic Press, 2012.

BENDSOE, M. P; SIGMUND, O. Topology Optimization - Theory, Methods and Applications. Berlin, **New York: Springer**, 2003

BOURDIN, B. Filters in topology optimization. International journal for numerical methods in engineering**. International journal for numerical methods in engineering**, v. 50, n. 9, p. 2143-2158, 2001.

COOK, R. D.; MALKUS, D. S.; PLESHA, M. E.; WITT, R. J. **Concepts and Applications of Finite Element Analysis** EUA: John Wiley e Sons, Inc. 2002.

FERRARI, F., SIGMUND, O. **A new generation 99-line MATLAB code for compliance topology optimization and its extension to 3D**. Struct Multidisc Optim 62, 2211–2228 (2020).

KIM, N.; SANKAR, B. **Introduction to Finite Element Analysis and design**. Ed. Livros Técnicos e Científicos. 2008.

LIU, K., TOVAR, A. **An efficient 3D topology optimization code written in MATLAB.** Structural and Multidisciplinary Optimization, 50.6, p 1175-1196. 2014.

MAIA, Sheyla M., BRITO, Thalyson I. J., COSTA, Jorge C. A SIMP-based algorithm to maximize natural frequencies in two-dimensional structures. **Proceedings of the XLII Ibero-Latin American Congress on Computational Methods in Engineering and 3rd Pan American Congress on Computational Mechanics**. ABMEC-IACM. Rio de Janeiro, Brazil, 2021.

MAXWELL, J. C. On Reciprocal Figures, Frames and Diagrams of Force. **Royal Society of Edinburgh**. 1870.

MEIJBOOM, Marion. **Topology optimization of dynamic problems**. Eindhoven University of Technology. 2003.

MICHELL, A. G. M. The limits of Economy of Material in Framed Structures. **The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science**. 1904.

NASCIMENTO, H.L.S. **Método de Otimização Topológica em Estruturas Contínuas Bidimensionais.** Trabalho de Conclusão de Curso – Universidade Federal do Rio Grande do Norte. Natal. 2019.

PEREIRA, R. E. L. **Otimização topológica de sistema de contraventamento em edificações, considerando os efeitos do vento**. Dissertação (Mestrado) –

Universidade Federal de Pernambuco, CAA, Programa de Pós-Graduação em Engenharia Civil e Ambiental. Caruaru, 2018.

PORTO, E.C.B. **Método da homogeneização aplicado à otimização estrutural topológica.** Dissertação de Mestrado, Universidade Estadual de Campinas – Faculdade de Engenharia Mecânica. Campinas. 2006.

REDDY, J. N. **An Introduction to the Finite Element Method**. 3. ed. Mc Graw Hill, 2006.

SIMONETTI, H. L. **Otimização topológica de estruturas bidimensionais**. Dissertação Mestrado – Universidade Federal de Ouro Preto. Escola de Minas. Departamento de Engenharia Civil. Programa de Pós-Graduação em Engenharia Civil. Ouro Preto. 2009.

SANT'ANNA, H. M. **Otimização Topológica de Estruturas Bidimensionais Contínuas submetidas a restrições de flexibilidade e tensão**. Dissertação de Mestrado - UFRGS. Porto Alegre. 2002.

SIGMUND, O. A 99-line topology optimization code written in MATLAB. **Springer Verlag**, Lyngby - Dinamarca, v. 21, n° 2, p.120–127, Abril, 2001.

SOSNOVIK, I., OSELEDETS, I. **Neural networks for topology optimization**. Russian Journal of Numerical Analysis and Mathematical Modelling, 2019.

VALENCIA, C. A. M. **Optimización topológica en el diseño de elementos estructurales mecánicos**. Santiago de Cali, 2012. p. 68. Monografía (Graduada em Engeñaría Mecánica) – Facultad de Ingeniería – Universidad Autónoma de Occidente.

ZHU, J.; ZHANG, W.; XIA, L. Topology optimization in aircraft and aerospace structures design. **Archives of Computational Methods in Engineering**, v. 23, n. 4, p. 595-622, 2016.

ANEXO A: CÓDIGO DE ANDRESSEN E CLAUSEN (TOP88)

05/05/23 13:41 C:\Users\Jarbas\D...\top88.m 1 of 4

```
%%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
function top88(nelx,nely,volfrac,penal,rmin,ft)
%% MATERIAL PROPERTIES
E0 = 1;
Emin = 1e-9;
nu = 0.3;
%% PREPARE FINITE ELEMENT ANALYSIS
A11 = [12  3 -6 -3;  3 12  3  0; -6  3 12 -3; -3  0 -3 12];
A12 = [-6 -3  0  3; -3 -6 -3 -6;  0 -3 -6  3;  3 -6  3 -6];
B11 = [-4  3 -2  9;  3 -4 -9  4; -2 -9 -4 -3;  9  4 -3 -4];
B12 = [ 2 -3  4 -9; -3  2  9 -2;  4  9  2  3; -9 -2  3  2];
KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12' B11]);
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],↙
nelx*nely,1);
iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
% DEFINE LOADS AND SUPPORTS (HALF MBB-BEAM)
F = sparse(2,1,-1,2*(nely+1)*(nelx+1),1);
U = zeros(2*(nely+1)*(nelx+1),1);
fixeddofs = union([1:2:2*(nely+1)],[2*(nelx+1)*(nely+1)]);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
%% PREPARE FILTER
iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
jH = ones(size(iH));
sH = zeros(size(iH));
k = 0;
for i1 = 1:nelx
  for j1 = 1:nely
    e1 = (i1-1)*nely+j1;
    for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
      for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),↙
nely)
        e2 = (i2-1)*nely+j2;
        k = k+1;
        iH(k) = e1;
        jH(k) = e2;
        sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
      end
    end
  end
end
```

```
 43 H = sparse(iH,jH,sH);
 44 Hs = sum(H,2);
 45 %% INITIALIZE ITERATION
 46 x = repmat(volfrac,nely,nelx);
 47 xPhys = x;
 48 loop = 0;
 49 change = 1;
 50 %% START ITERATION
 51 while change > 0.01
 52   loop = loop + 1;
 53   %% FE-ANALYSIS
 54   sK = reshape(KE(:)*(Emin+xPhys(:)'.^penal*(E0-Emin)),↙
64*nelx*nely,1);
 55   K = sparse(iK,jK,sK); K = (K+K')/2;
 56   U(freedofs) = K(freedofs,freedofs)\F(freedofs);
 57   %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
 58   ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
 59   c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce));
 60   dc = -penal*(E0-Emin)*xPhys.^(penal-1).*ce;
 61   dv = ones(nely,nelx);
 62   %% FILTERING/MODIFICATION OF SENSITIVITIES
 63   if ft == 1
 64     dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
 65   elseif ft == 2
 66     dc(:) = H*(dc(:)./Hs);
 67     dv(:) = H*(dv(:)./Hs);
 68   end
 69   %% OPTIMALITY CRITERIA UPDATE OF DESIGN VARIABLES AND PHYSICAL↙
DENSITIES
 70   l1 = 0; l2 = 1e9; move = 0.2;
 71   while (l2-l1)/(l1+l2) > 1e-3
 72     lmid = 0.5*(l2+l1);
 73     xnew = max(0,max(x-move,min(1,min(x+move,x.*sqrt(-dc.↙
/dv/lmid)))));
 74     if ft == 1
 75       xPhys = xnew;
 76     elseif ft == 2
 77       xPhys(:) = (H*xnew(:))./Hs;
 78     end
 79     if sum(xPhys(:)) > volfrac*nelx*nely, l1 = lmid; else l2 =↙
lmid; end
 80   end
 81   change = max(abs(xnew(:)-x(:)));
 82   x = xnew;
```

```
 83   %% PRINT RESULTS
 84   fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c,↙
...
 85     mean(xPhys(:)),change);
 86   %% PLOT DENSITIES
 87   colormap(gray); imagesc(1-xPhys); caxis([0 1]); axis equal; axis↙
off; drawnow;
 88 end
 89 %
 90↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
 91 % This Matlab code was written by E. Andreassen, A. Clausen, M.↙
Schevenels,%
 92 % B. S. Lazarov and O. Sigmund,  Department of Solid  Mechanics,↙
%
 93 %  Technical University of Denmark,↙
%
 94 %  DK-2800 Lyngby, Denmark.↙
%
 95 % Please sent your comments to: sigmund@fam.dtu.dk↙
%
 96 %↙
%
 97 % The code is intended for educational purposes and theoretical↙
details    %
 98 % are discussed in the paper↙
%
 99 % "Efficient topology optimization in MATLAB using 88 lines of↙
code,       %
100 % E. Andreassen, A. Clausen, M. Schevenels,↙
%
101 % B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010↙
%
102 % This version is based on earlier 99-line code↙
%
103 % by Ole Sigmund (2001), Structural and Multidisciplinary↙
Optimization,    %
104 % Vol 21, pp. 120--127.↙
%
105 %↙
%
106 % The code as well as a postscript version of the paper can be↙
%
```

```
107 % downloaded from the web-site: http://www.topopt.dtu.dk↙
%
108 %↙
%
109 % Disclaimer:↙
%
110 % The authors reserves all rights but do not guaranty that the↙
code is     %
111 % free from errors. Furthermore, we shall not be liable in any↙
event       %
112 % caused by the use of the program.↙
%
113↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
114
115
```

ANEXO B: CÓDIGO DE ANDRESSEN E CLAUSEN ADAPTADO PARA O CASO 1 – VIGA BIAPOIADA

05/05/23 13:37 C:\U...\Vigabiapoiadafundo.m 1 of 4

```
%%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
function top88(nelx,nely,volfrac,penal,rmin,ft)
%% MATERIAL PROPERTIES
E0 = 1;
Emin = 1e-9;
nu = 0.3;
%% PREPARE FINITE ELEMENT ANALYSIS
A11 = [12  3 -6 -3;  3 12  3  0; -6  3 12 -3; -3  0 -3 12];
A12 = [-6 -3  0  3; -3 -6 -3 -6;  0 -3 -6  3;  3 -6  3 -6];
B11 = [-4  3 -2  9;  3 -4 -9  4; -2 -9 -4 -3;  9  4 -3 -4];
B12 = [ 2 -3  4 -9; -3  2  9 -2;  4  9  2  3; -9 -2  3  2];
KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12' B11]);
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],↙
nelx*nely,1);
iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
% DEFINE LOADS AND SUPPORTS
F = sparse((nely+1)*nelx,1,-1,2*(nely+1)*(nelx+1),1);
U = zeros(2*(nely+1)*(nelx+1),1);
fixeddofs = union([2*(nely+1)],[2*(nely+1)*(nelx+1)]);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
%% PREPARE FILTER
iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
jH = ones(size(iH));
sH = zeros(size(iH));
k = 0;
for i1 = 1:nelx
  for j1 = 1:nely
    e1 = (i1-1)*nely+j1;
    for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
      for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),↙
nely)
        e2 = (i2-1)*nely+j2;
        k = k+1;
        iH(k) = e1;
        jH(k) = e2;
        sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
      end
    end
  end
end
```

```
 43 H = sparse(iH,jH,sH);
 44 Hs = sum(H,2);
 45 %% INITIALIZE ITERATION
 46 x = repmat(volfrac,nely,nelx);
 47 xPhys = x;
 48 loop = 0;
 49 change = 1;
 50 %% START ITERATION
 51 while change > 0.01
 52   loop = loop + 1;
 53   %% FE-ANALYSIS
 54   sK = reshape(KE(:)*(Emin+xPhys(:)'.^penal*(E0-Emin)),↙
64*nelx*nely,1);
 55   K = sparse(iK,jK,sK); K = (K+K')/2;
 56   U(freedofs) = K(freedofs,freedofs)\F(freedofs);
 57   %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
 58   ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
 59   c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce));
 60   dc = -penal*(E0-Emin)*xPhys.^(penal-1).*ce;
 61   dv = ones(nely,nelx);
 62   %% FILTERING/MODIFICATION OF SENSITIVITIES
 63   if ft == 1
 64     dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
 65   elseif ft == 2
 66     dc(:) = H*(dc(:)./Hs);
 67     dv(:) = H*(dv(:)./Hs);
 68   end
 69   %% OPTIMALITY CRITERIA UPDATE OF DESIGN VARIABLES AND PHYSICAL↙
DENSITIES
 70   l1 = 0; l2 = 1e9; move = 0.2;
 71   while (l2-l1)/(l1+l2) > 1e-3
 72     lmid = 0.5*(l2+l1);
 73     xnew = max(0,max(x-move,min(1,min(x+move,x.*sqrt(-dc.↙
/dv/lmid)))));
 74     if ft == 1
 75       xPhys = xnew;
 76     elseif ft == 2
 77       xPhys(:) = (H*xnew(:))./Hs;
 78     end
 79     if sum(xPhys(:)) > volfrac*nelx*nely, l1 = lmid; else l2 =↙
lmid; end
 80   end
 81   change = max(abs(xnew(:)-x(:)));
 82   x = xnew;
```

```
 83   %% PRINT RESULTS
 84   fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c,↙
...
 85     mean(xPhys(:)),change);
 86   %% PLOT DENSITIES
 87   colormap(gray); imagesc(1-xPhys); caxis([0 1]); axis equal; axis↙
off; drawnow;
 88 end
 89 %
 90↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
 91 % This Matlab code was written by E. Andreassen, A. Clausen, M.↙
Schevenels,%
 92 % B. S. Lazarov and O. Sigmund,  Department of Solid  Mechanics,↙
%
 93 %  Technical University of Denmark,↙
%
 94 %  DK-2800 Lyngby, Denmark.↙
%
 95 % Please sent your comments to: sigmund@fam.dtu.dk↙
%
 96 %↙
%
 97 % The code is intended for educational purposes and theoretical↙
details    %
 98 % are discussed in the paper↙
%
 99 % "Efficient topology optimization in MATLAB using 88 lines of↙
code,       %
100 % E. Andreassen, A. Clausen, M. Schevenels,↙
%
101 % B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010↙
%
102 % This version is based on earlier 99-line code↙
%
103 % by Ole Sigmund (2001), Structural and Multidisciplinary↙
Optimization,    %
104 % Vol 21, pp. 120--127.↙
%
105 %↙
%
106 % The code as well as a postscript version of the paper can be↙
%
```

```
107 % downloaded from the web-site: http://www.topopt.dtu.dk↙
%
108 %↙
%
109 % Disclaimer:↙
%
110 % The authors reserves all rights but do not guaranty that the↙
code is     %
111 % free from errors. Furthermore, we shall not be liable in any↙
event       %
112 % caused by the use of the program.↙
%
113↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
114
115
```

ANEXO C: CÓDIGO DE ANDRESSEN E CLAUSEN ADAPTADO PARA O CASO 2 – VIGA EM BALANÇO


```
  1 %%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
  2 function top88(nelx,nely,volfrac,penal,rmin,ft)
  3 %% MATERIAL PROPERTIES
  4 E0 = 1;
  5 Emin = 1e-9;
  6 nu = 0.3;
  7 %% PREPARE FINITE ELEMENT ANALYSIS
  8 A11 = [12  3 -6 -3;  3 12  3  0; -6  3 12 -3; -3  0 -3 12];
  9 A12 = [-6 -3  0  3; -3 -6 -3 -6;  0 -3 -6  3;  3 -6  3 -6];
 10 B11 = [-4  3 -2  9;  3 -4 -9  4; -2 -9 -4 -3;  9  4 -3 -4];
 11 B12 = [ 2 -3  4 -9; -3  2  9 -2;  4  9  2  3; -9 -2  3  2];
 12 KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12' B11]);
 13 nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
 14 edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
 15 edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],↙
nelx*nely,1);
 16 iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
 17 jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
 18 % DEFINE LOADS AND SUPPORTS
 19 F = sparse(2*(nely+1)*(nelx+1),1,-1,2*(nely+1)*(nelx+1),1);
 20 U = zeros(2*(nely+1)*(nelx+1),1);
 21 fixeddofs = union(1,[2:(2*(nely+1))]);
 22 alldofs = [1:2*(nely+1)*(nelx+1)];
 23 freedofs = setdiff(alldofs,fixeddofs);
 24 %% PREPARE FILTER
 25 iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
 26 jH = ones(size(iH));
 27 sH = zeros(size(iH));
 28 k = 0;
 29 for i1 = 1:nelx
 30   for j1 = 1:nely
 31     e1 = (i1-1)*nely+j1;
 32     for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
 33       for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),↙
nely)
 34         e2 = (i2-1)*nely+j2;
 35         k = k+1;
 36         iH(k) = e1;
 37         jH(k) = e2;
 38         sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
 39       end
 40     end
 41   end
 42 end
```

```
 43 H = sparse(iH,jH,sH);
 44 Hs = sum(H,2);
 45 %% INITIALIZE ITERATION
 46 x = repmat(volfrac,nely,nelx);
 47 xPhys = x;
 48 loop = 0;
 49 change = 1;
 50 %% START ITERATION
 51 while change > 0.01
 52   loop = loop + 1;
 53   %% FE-ANALYSIS
 54   sK = reshape(KE(:)*(Emin+xPhys(:)'.^penal*(E0-Emin)),↙
64*nelx*nely,1);
 55   K = sparse(iK,jK,sK); K = (K+K')/2;
 56   U(freedofs) = K(freedofs,freedofs)\F(freedofs);
 57   %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
 58   ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
 59   c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce));
 60   dc = -penal*(E0-Emin)*xPhys.^(penal-1).*ce;
 61   dv = ones(nely,nelx);
 62   %% FILTERING/MODIFICATION OF SENSITIVITIES
 63   if ft == 1
 64     dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
 65   elseif ft == 2
 66     dc(:) = H*(dc(:)./Hs);
 67     dv(:) = H*(dv(:)./Hs);
 68   end
 69   %% OPTIMALITY CRITERIA UPDATE OF DESIGN VARIABLES AND PHYSICAL↙
DENSITIES
 70   l1 = 0; l2 = 1e9; move = 0.2;
 71   while (l2-l1)/(l1+l2) > 1e-3
 72     lmid = 0.5*(l2+l1);
 73     xnew = max(0,max(x-move,min(1,min(x+move,x.*sqrt(-dc.↙
/dv/lmid)))));
 74     if ft == 1
 75       xPhys = xnew;
 76     elseif ft == 2
 77       xPhys(:) = (H*xnew(:))./Hs;
 78     end
 79     if sum(xPhys(:)) > volfrac*nelx*nely, l1 = lmid; else l2 =↙
lmid; end
 80   end
 81   change = max(abs(xnew(:)-x(:)));
 82   x = xnew;
```

```
 83   %% PRINT RESULTS
 84   fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c,...
 85     mean(xPhys(:)),change);
 86   %% PLOT DENSITIES
 87   colormap(gray); imagesc(1-xPhys); caxis([0 1]); axis equal; axis off; drawnow;
 88 end
 89 %
 90
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 91 % This Matlab code was written by E. Andreassen, A. Clausen, M. Schevenels,%
 92 % B. S. Lazarov and O. Sigmund,  Department of Solid  Mechanics,
%
 93 %  Technical University of Denmark,
%
 94 %  DK-2800 Lyngby, Denmark.
%
 95 % Please sent your comments to: sigmund@fam.dtu.dk
%
 96 %
%
 97 % The code is intended for educational purposes and theoretical details    %
 98 % are discussed in the paper
%
 99 % "Efficient topology optimization in MATLAB using 88 lines of code,       %
100 % E. Andreassen, A. Clausen, M. Schevenels,
%
101 % B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010
%
102 % This version is based on earlier 99-line code
%
103 % by Ole Sigmund (2001), Structural and Multidisciplinary Optimization,    %
104 % Vol 21, pp. 120--127.
%
105 %
%
106 % The code as well as a postscript version of the paper can be
%
```

```
107 % downloaded from the web-site: http://www.topopt.dtu.dk↙
%
108 %↙
%
109 % Disclaimer:↙
%
110 % The authors reserves all rights but do not guaranty that the↙
code is     %
111 % free from errors. Furthermore, we shall not be liable in any↙
event       %
112 % caused by the use of the program.↙
%
113↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
114
115
```

ANEXO D: CÓDIGO DE ANDRESSEN E CLAUSEN ADAPTADO PARA O CASO 3 – VIGA DE MICHELL

05/05/23 13:38 C:\Users\Ja...\VigaMichell.m 1 of 4

```
%%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
function top88(nelx,nely,volfrac,penal,rmin,ft)
%% MATERIAL PROPERTIES
E0 = 1;
Emin = 1e-9;
nu = 0.3;
%% PREPARE FINITE ELEMENT ANALYSIS
A11 = [12  3 -6 -3;  3 12  3  0; -6  3 12 -3; -3  0 -3 12];
A12 = [-6 -3  0  3; -3 -6 -3 -6;  0 -3 -6  3;  3 -6  3 -6];
B11 = [-4  3 -2  9;  3 -4 -9  4; -2 -9 -4 -3;  9  4 -3 -4];
B12 = [ 2 -3  4 -9; -3  2  9 -2;  4  9  2  3; -9 -2  3  2];
KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12' B11]);
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],↙
nelx*nely,1);
iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
% DEFINE LOADS AND SUPPORTS
F = sparse(2*(nely)*((nelx/2)),1,-1,2*(nely+1)*(nelx+1),1);
U = zeros(2*(nely+1)*(nelx+1),1);
fixeddofs = union([2*(nely+1)],[2*(nely+1)*(nelx+1)]);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
%% PREPARE FILTER
iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
jH = ones(size(iH));
sH = zeros(size(iH));
k = 0;
for i1 = 1:nelx
  for j1 = 1:nely
    e1 = (i1-1)*nely+j1;
    for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
      for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),↙
nely)
        e2 = (i2-1)*nely+j2;
        k = k+1;
        iH(k) = e1;
        jH(k) = e2;
        sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
      end
    end
  end
end
```

```
 43 H = sparse(iH,jH,sH);
 44 Hs = sum(H,2);
 45 %% INITIALIZE ITERATION
 46 x = repmat(volfrac,nely,nelx);
 47 xPhys = x;
 48 loop = 0;
 49 change = 1;
 50 %% START ITERATION
 51 while change > 0.01
 52   loop = loop + 1;
 53   %% FE-ANALYSIS
 54   sK = reshape(KE(:)*(Emin+xPhys(:)'.^penal*(E0-Emin)),↙
64*nelx*nely,1);
 55   K = sparse(iK,jK,sK); K = (K+K')/2;
 56   U(freedofs) = K(freedofs,freedofs)\F(freedofs);
 57   %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
 58   ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
 59   c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce));
 60   dc = -penal*(E0-Emin)*xPhys.^(penal-1).*ce;
 61   dv = ones(nely,nelx);
 62   %% FILTERING/MODIFICATION OF SENSITIVITIES
 63   if ft == 1
 64     dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
 65   elseif ft == 2
 66     dc(:) = H*(dc(:)./Hs);
 67     dv(:) = H*(dv(:)./Hs);
 68   end
 69   %% OPTIMALITY CRITERIA UPDATE OF DESIGN VARIABLES AND PHYSICAL↙
DENSITIES
 70   l1 = 0; l2 = 1e9; move = 0.2;
 71   while (l2-l1)/(l1+l2) > 1e-3
 72     lmid = 0.5*(l2+l1);
 73     xnew = max(0,max(x-move,min(1,min(x+move,x.*sqrt(-dc.↙
/dv/lmid)))));
 74     if ft == 1
 75       xPhys = xnew;
 76     elseif ft == 2
 77       xPhys(:) = (H*xnew(:))./Hs;
 78     end
 79     if sum(xPhys(:)) > volfrac*nelx*nely, l1 = lmid; else l2 =↙
lmid; end
 80   end
 81   change = max(abs(xnew(:)-x(:)));
 82   x = xnew;
```

```
 83   %% PRINT RESULTS
 84   fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c,↙
...
 85     mean(xPhys(:)),change);
 86   %% PLOT DENSITIES
 87   colormap(gray); imagesc(1-xPhys); caxis([0 1]); axis equal; axis↙
off; drawnow;
 88 end
 89 %
 90↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
 91 % This Matlab code was written by E. Andreassen, A. Clausen, M.↙
Schevenels,%
 92 % B. S. Lazarov and O. Sigmund,  Department of Solid  Mechanics,↙
%
 93 %  Technical University of Denmark,↙
%
 94 %  DK-2800 Lyngby, Denmark.↙
%
 95 % Please sent your comments to: sigmund@fam.dtu.dk↙
%
 96 %↙
%
 97 % The code is intended for educational purposes and theoretical↙
details    %
 98 % are discussed in the paper↙
%
 99 % "Efficient topology optimization in MATLAB using 88 lines of↙
code,       %
100 % E. Andreassen, A. Clausen, M. Schevenels,↙
%
101 % B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010↙
%
102 % This version is based on earlier 99-line code↙
%
103 % by Ole Sigmund (2001), Structural and Multidisciplinary↙
Optimization,    %
104 % Vol 21, pp. 120--127.↙
%
105 %↙
%
106 % The code as well as a postscript version of the paper can be↙
%
```

```
107 % downloaded from the web-site: http://www.topopt.dtu.dk↙
%
108 %↙
%
109 % Disclaimer:↙
%
110 % The authors reserves all rights but do not guaranty that the↙
code is     %
111 % free from errors. Furthermore, we shall not be liable in any↙
event       %
112 % caused by the use of the program.↙
%
113↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
114
115
```

ANEXO E: CÓDIGO DE ANDRESSEN E CLAUSEN ADAPTADO PARA O CASO 4 –VIGA BIENGASTADA

05/05/23 13:38 C:\User...\VigaBiEngastada.m 1 of 4

```
  1 %%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
  2 function top88(nelx,nely,volfrac,penal,rmin,ft)
  3 %% MATERIAL PROPERTIES
  4 E0 = 1;
  5 Emin = 1e-9;
  6 nu = 0.3;
  7 %% PREPARE FINITE ELEMENT ANALYSIS
  8 A11 = [12  3 -6 -3;  3 12  3  0; -6  3 12 -3; -3  0 -3 12];
  9 A12 = [-6 -3  0  3; -3 -6 -3 -6;  0 -3 -6  3;  3 -6  3 -6];
 10 B11 = [-4  3 -2  9;  3 -4 -9  4; -2 -9 -4 -3;  9  4 -3 -4];
 11 B12 = [ 2 -3  4 -9; -3  2  9 -2;  4  9  2  3; -9 -2  3  2];
 12 KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12' B11]);
 13 nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
 14 edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
 15 edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],↙
nelx*nely,1);
 16 iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
 17 jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
 18 % DEFINE LOADS AND SUPPORTS
 19 F = sparse(2*(nely)*((nelx/2)),1,-1,2*(nely+1)*(nelx+1),1);
 20 U = zeros(2*(nely+1)*(nelx+1),1);
 21 fixeddofs = union([1:2*(nely+1)],[(2*(nely+1)*(nelx+1)-(2*(nely+1)↙
-1):2*(nely+1)*(nelx+1))]);
 22 alldofs = [1:2*(nely+1)*(nelx+1)];
 23 freedofs = setdiff(alldofs,fixeddofs);
 24 %% PREPARE FILTER
 25 iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
 26 jH = ones(size(iH));
 27 sH = zeros(size(iH));
 28 k = 0;
 29 for i1 = 1:nelx
 30   for j1 = 1:nely
 31     e1 = (i1-1)*nely+j1;
 32     for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
 33       for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),↙
nely)
 34         e2 = (i2-1)*nely+j2;
 35         k = k+1;
 36         iH(k) = e1;
 37         jH(k) = e2;
 38         sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
 39       end
 40     end
 41   end
```

```
 42 end
 43 H = sparse(iH,jH,sH);
 44 Hs = sum(H,2);
 45 %% INITIALIZE ITERATION
 46 x = repmat(volfrac,nely,nelx);
 47 xPhys = x;
 48 loop = 0;
 49 change = 1;
 50 %% START ITERATION
 51 while change > 0.01
 52   loop = loop + 1;
 53   %% FE-ANALYSIS
 54   sK = reshape(KE(:)*(Emin+xPhys(:)'.^penal*(E0-Emin)),↙
64*nelx*nely,1);
 55   K = sparse(iK,jK,sK); K = (K+K')/2;
 56   U(freedofs) = K(freedofs,freedofs)\F(freedofs);
 57   %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
 58   ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
 59   c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce));
 60   dc = -penal*(E0-Emin)*xPhys.^(penal-1).*ce;
 61   dv = ones(nely,nelx);
 62   %% FILTERING/MODIFICATION OF SENSITIVITIES
 63   if ft == 1
 64     dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
 65   elseif ft == 2
 66     dc(:) = H*(dc(:)./Hs);
 67     dv(:) = H*(dv(:)./Hs);
 68   end
 69   %% OPTIMALITY CRITERIA UPDATE OF DESIGN VARIABLES AND PHYSICAL↙
DENSITIES
 70   l1 = 0; l2 = 1e9; move = 0.2;
 71   while (l2-l1)/(l1+l2) > 1e-3
 72     lmid = 0.5*(l2+l1);
 73     xnew = max(0,max(x-move,min(1,min(x+move,x.*sqrt(-dc.↙
/dv/lmid)))));
 74     if ft == 1
 75       xPhys = xnew;
 76     elseif ft == 2
 77       xPhys(:) = (H*xnew(:))./Hs;
 78     end
 79     if sum(xPhys(:)) > volfrac*nelx*nely, l1 = lmid; else l2 =↙
lmid; end
 80   end
 81   change = max(abs(xnew(:)-x(:)));
```

```
  x = xnew;
  %% PRINT RESULTS
  fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c,↙
...
    mean(xPhys(:)),change);
  %% PLOT DENSITIES
  colormap(gray); imagesc(1-xPhys); caxis([0 1]); axis equal; axis↙
off; drawnow;
end
%
↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
% This Matlab code was written by E. Andreassen, A. Clausen, M.↙
Schevenels,%
% B. S. Lazarov and O. Sigmund,  Department of Solid  Mechanics,↙
%
%  Technical University of Denmark,↙
%
%  DK-2800 Lyngby, Denmark.↙
%
% Please sent your comments to: sigmund@fam.dtu.dk↙
%
%↙
%
% The code is intended for educational purposes and theoretical↙
details    %
% are discussed in the paper↙
%
% "Efficient topology optimization in MATLAB using 88 lines of↙
code,       %
% E. Andreassen, A. Clausen, M. Schevenels,↙
%
% B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010↙
%
% This version is based on earlier 99-line code↙
%
% by Ole Sigmund (2001), Structural and Multidisciplinary↙
Optimization,    %
% Vol 21, pp. 120--127.↙
%
%↙
%
% The code as well as a postscript version of the paper can be↙
```

```
%
107 % downloaded from the web-site: http://www.topopt.dtu.dk↙
%
108 %↙
%
109 % Disclaimer:↙
%
110 % The authors reserves all rights but do not guaranty that the↙
code is     %
111 % free from errors. Furthermore, we shall not be liable in any↙
event       %
112 % caused by the use of the program.↙
%
113↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
114
115
```

ANEXO F: CÓDIGO DE ANDRESSEN E CLAUSEN ADAPTADO PARA O CASO 5 – PONTE COM TABULEIRO SUPERIOR

17/05/23 13:42 C:\Users\Jarbas\O...\Ponte.m 1 of 4

```
%%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
function top88(nelx,nely,volfrac,penal,rmin,ft)
%% MATERIAL PROPERTIES
E0 = 1;
Emin = 1e-9;
nu = 0.3;
%% PREPARE FINITE ELEMENT ANALYSIS
A11 = [12  3 -6 -3;  3 12  3  0; -6  3 12 -3; -3  0 -3 12];
A12 = [-6 -3  0  3; -3 -6 -3 -6;  0 -3 -6  3;  3 -6  3 -6];
B11 = [-4  3 -2  9;  3 -4 -9  4; -2 -9 -4 -3;  9  4 -3 -4];
B12 = [ 2 -3  4 -9; -3  2  9 -2;  4  9  2  3; -9 -2  3  2];
KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12' B11]);
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],↙
nelx*nely,1);
iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
% DEFINE LOADS AND SUPPORTS
for i=1:(nelx+1)
    tab = 2*(nely+1);
    cent = (nely+1)+(i-1)*tab+1;
    vet1 (i,1) = 2+(i-1)*tab;
    vet2 (i,1) = vet1 (i,1) - 2;
    vet3 (i,1) = vet1 (i,1) + 2;
    vet4 (i,1) = vet2 (i,1) + 2;
    vet5 (i,1) = vet2 (i,1) - 1;
end
Aux1 = ((2*(nelx+1)*(nely+1))*(1/4)) - 0.5;
Aux2 = (2*(nelx+1)*(nely+1))*(2/4);
Aux3 = (((2*(nelx+1)*(nely+1))*(3/4)) + 0.5);
F = sparse(vet1,1,-1,2*(nely+1)*(nelx+1),1);
U = zeros(2*(nely+1)*(nelx+1),1);
fixeddofs = union(vet2,vet5);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
%% PREPARE FILTER
iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
jH = ones(size(iH));
sH = zeros(size(iH));
k = 0;
for i1 = 1:nelx
  for j1 = 1:nely
    e1 = (i1-1)*nely+j1;
```

```
 44      for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
 45        for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),↙
nely)
 46          e2 = (i2-1)*nely+j2;
 47          k = k+1;
 48          iH(k) = e1;
 49          jH(k) = e2;
 50          sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
 51        end
 52      end
 53    end
 54  end
 55  H = sparse(iH,jH,sH);
 56  Hs = sum(H,2);
 57  %% INITIALIZE ITERATION
 58  x = repmat(volfrac,nely,nelx);
 59  xPhys = x;
 60  loop = 0;
 61  change = 1;
 62  %% START ITERATION
 63  while change > 0.01
 64    loop = loop + 1;
 65    %% FE-ANALYSIS
 66    sK = reshape(KE(:)*(Emin+xPhys(:)'.^penal*(E0-Emin)),↙
64*nelx*nely,1);
 67    K = sparse(iK,jK,sK); K = (K+K')/2;
 68    U(freedofs) = K(freedofs,freedofs)\F(freedofs);
 69    %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
 70    ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
 71    c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce));
 72    dc = -penal*(E0-Emin)*xPhys.^(penal-1).*ce;
 73    dv = ones(nely,nelx);
 74    %% FILTERING/MODIFICATION OF SENSITIVITIES
 75    if ft == 1
 76      dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
 77    elseif ft == 2
 78      dc(:) = H*(dc(:)./Hs);
 79      dv(:) = H*(dv(:)./Hs);
 80    end
 81    %% OPTIMALITY CRITERIA UPDATE OF DESIGN VARIABLES AND PHYSICAL↙
DENSITIES
 82    l1 = 0; l2 = 1e9; move = 0.2;
 83    while (l2-l1)/(l1+l2) > 1e-3
 84      lmid = 0.5*(l2+l1);
```

```
 85     xnew = max(0,max(x-move,min(1,min(x+move,x.*sqrt(-dc.↙
/dv/lmid))));
 86     if ft == 1
 87       xPhys = xnew;
 88     elseif ft == 2
 89       xPhys(:) = (H*xnew(:))./Hs;
 90     end
 91     if sum(xPhys(:)) > volfrac*nelx*nely, l1 = lmid; else l2 =↙
lmid; end
 92   end
 93   change = max(abs(xnew(:)-x(:)));
 94   x = xnew;
 95   %% PRINT RESULTS
 96   fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c,↙
...
 97     mean(xPhys(:)),change);
 98   %% PLOT DENSITIES
 99   colormap(gray); imagesc(1-xPhys); caxis([0 1]); axis equal; axis↙
off; drawnow;
100 end
101 %
102↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
103 % This Matlab code was written by E. Andreassen, A. Clausen, M.↙
Schevenels,%
104 % B. S. Lazarov and O. Sigmund,  Department of Solid  Mechanics,↙
%
105 %  Technical University of Denmark,↙
%
106 %  DK-2800 Lyngby, Denmark.↙
%
107 % Please sent your comments to: sigmund@fam.dtu.dk↙
%
108 %↙
%
109 % The code is intended for educational purposes and theoretical↙
details    %
110 % are discussed in the paper↙
%
111 % "Efficient topology optimization in MATLAB using 88 lines of↙
code,       %
112 % E. Andreassen, A. Clausen, M. Schevenels,↙
%
```

```
113 % B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010↙
%
114 % This version is based on earlier 99-line code↙
%
115 % by Ole Sigmund (2001), Structural and Multidisciplinary↙
Optimization,    %
116 % Vol 21, pp. 120--127.↙
%
117 %↙
%
118 % The code as well as a postscript version of the paper can be↙
%
119 % downloaded from the web-site: http://www.topopt.dtu.dk↙
%
120 %↙
%
121 % Disclaimer:↙
%
122 % The authors reserves all rights but do not guaranty that the↙
code is     %
123 % free from errors. Furthermore, we shall not be liable in any↙
event       %
124 % caused by the use of the program.↙
%
125↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
126
127
```

ANEXO G: CÓDIGO DE ANDRESSEN E CLAUSEN ADAPTADO PARA O CASO 6 – PRÉDIO

17/05/23 15:53 C:\Users\Jarbas\...\Predio.m 1 of 4

```
%%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
function top88(nelx,nely,volfrac,penal,rmin,ft)
%% MATERIAL PROPERTIES
E0 = 1;
Emin = 1e-9;
nu = 0.3;
%% PREPARE FINITE ELEMENT ANALYSIS
A11 = [12  3 -6 -3;  3 12  3  0; -6  3 12 -3; -3  0 -3 12];
A12 = [-6 -3  0  3; -3 -6 -3 -6;  0 -3 -6  3;  3 -6  3 -6];
B11 = [-4  3 -2  9;  3 -4 -9  4; -2 -9 -4 -3;  9  4 -3 -4];
B12 = [ 2 -3  4 -9; -3  2  9 -2;  4  9  2  3; -9 -2  3  2];
KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12' B11]);
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],↙
nelx*nely,1);
iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
% DEFINE LOADS AND SUPPORTS
tab = 2*(nely+1);
for i=1:(nelx+1)
    cent = (nely+1)+(i-1)*tab+1;
    vet1 (i,1) = 2+(i-1)*tab;
    vet2 (i,1) = vet1 (i,1) - 2;
    vet3 (i,1) = (2*i)-1;
    vet4 (i,1) = vet2 (i,1) - 1;
    vet5 (i,1) = vet1 (i,1) + vet3 (i,1);
end
F = sparse(vet5,1,1,2*(nely+1)*(nelx+1),1);
U = zeros(2*(nely+1)*(nelx+1),1);
fixeddofs = union(vet2,vet4);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
%% PREPARE FILTER
iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
jH = ones(size(iH));
sH = zeros(size(iH));
k = 0;
for i1 = 1:nelx
  for j1 = 1:nely
    e1 = (i1-1)*nely+j1;
    for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
      for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),↙
nely)
```

```
 43          e2 = (i2-1)*nely+j2;
 44          k = k+1;
 45          iH(k) = e1;
 46          jH(k) = e2;
 47          sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
 48        end
 49      end
 50    end
 51  end
 52  H = sparse(iH,jH,sH);
 53  Hs = sum(H,2);
 54  %% INITIALIZE ITERATION
 55  x = repmat(volfrac,nely,nelx);
 56  xPhys = x;
 57  loop = 0;
 58  change = 1;
 59  %% START ITERATION
 60  while change > 0.01
 61    loop = loop + 1;
 62    %% FE-ANALYSIS
 63    sK = reshape(KE(:)*(Emin+xPhys(:)'.^penal*(E0-Emin)),↙
64*nelx*nely,1);
 64    K = sparse(iK,jK,sK); K = (K+K')/2;
 65    U(freedofs) = K(freedofs,freedofs)\F(freedofs);
 66    %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
 67    ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
 68    c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce));
 69    dc = -penal*(E0-Emin)*xPhys.^(penal-1).*ce;
 70    dv = ones(nely,nelx);
 71    %% FILTERING/MODIFICATION OF SENSITIVITIES
 72    if ft == 1
 73      dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
 74    elseif ft == 2
 75      dc(:) = H*(dc(:)./Hs);
 76      dv(:) = H*(dv(:)./Hs);
 77    end
 78    %% OPTIMALITY CRITERIA UPDATE OF DESIGN VARIABLES AND PHYSICAL↙
DENSITIES
 79    l1 = 0; l2 = 1e9; move = 0.2;
 80    while (l2-l1)/(l1+l2) > 1e-3
 81      lmid = 0.5*(l2+l1);
 82      xnew = max(0,max(x-move,min(1,min(x+move,x.*sqrt(-dc.↙
/dv/lmid)))));
 83      if ft == 1
```

```
 84       xPhys = xnew;
 85     elseif ft == 2
 86       xPhys(:) = (H*xnew(:))./Hs;
 87     end
 88     if sum(xPhys(:)) > volfrac*nelx*nely, l1 = lmid; else l2 =↙
lmid; end
 89   end
 90   change = max(abs(xnew(:)-x(:)));
 91   x = xnew;
 92   %% PRINT RESULTS
 93   fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c,↙
...
 94     mean(xPhys(:)),change);
 95   %% PLOT DENSITIES
 96   colormap(gray); imagesc(1-xPhys); caxis([0 1]); axis equal; axis↙
off; drawnow;
 97 end
 98 %
 99↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
100 % This Matlab code was written by E. Andreassen, A. Clausen, M.↙
Schevenels,%
101 % B. S. Lazarov and O. Sigmund,  Department of Solid  Mechanics,↙
%
102 %  Technical University of Denmark,↙
%
103 %  DK-2800 Lyngby, Denmark.↙
%
104 % Please sent your comments to: sigmund@fam.dtu.dk↙
%
105 %↙
%
106 % The code is intended for educational purposes and theoretical↙
details    %
107 % are discussed in the paper↙
%
108 % "Efficient topology optimization in MATLAB using 88 lines of↙
code,       %
109 % E. Andreassen, A. Clausen, M. Schevenels,↙
%
110 % B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010↙
%
111 % This version is based on earlier 99-line code↙
```

```
%
112 % by Ole Sigmund (2001), Structural and Multidisciplinary↙
Optimization,    %
113 % Vol 21, pp. 120--127.↙
%
114 %↙
%
115 % The code as well as a postscript version of the paper can be↙
%
116 % downloaded from the web-site: http://www.topopt.dtu.dk↙
%
117 %↙
%
118 % Disclaimer:↙
%
119 % The authors reserves all rights but do not guaranty that the↙
code is     %
120 % free from errors. Furthermore, we shall not be liable in any↙
event       %
121 % caused by the use of the program.↙
%
122↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%↙
%%%%%%
123
124
```