

Karina Assis Rocha Yamamoto

**Otimização estrutural de pórticos
planos com o uso do algoritmo *SGA*
e análise não-linear**

Florianópolis

2015

Karina Assis Rocha Yamamoto

**Otimização estrutural de pórticos planos
com o uso do algoritmo *SGA* e análise
não-linear**

Trabalho de conclusão de curso
submetido ao Departamento de
Engenharia Civil da Universidade
Federal de Santa Catarina para a
obtenção do título de Engenheira Civil

Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Engenharia Civil

Orientador: Prof. Dr. Rafael Holdorf Lopez

Florianópolis
2015

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Yamamoto, Karina Assis Rocha
Otimização estrutural de pórticos planos com o uso do
algoritmo SGA e análise não-linear / Karina Assis Rocha
Yamamoto ; orientador, Rafael Holdorf Lopez -
Florianópolis, SC, 2015.
120 p.

Trabalho de Conclusão de Curso (graduação) -
Universidade Federal de Santa Catarina, Centro Tecnológico.
Graduação em Engenharia Civil.

Inclui referências

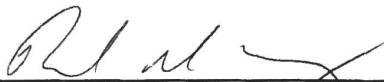
1. Engenharia Civil. 2. Otimização estrutural. 3.
Pórtico. 4. SGA. 5. Análise não-linear. I. Lopez, Rafael
Holdorf . II. Universidade Federal de Santa Catarina.
Graduação em Engenharia Civil. III. Título.

Karina Assis Rocha Yamamoto

Otimização estrutural de pórticos planos com o uso do algoritmo *SGA* e análise não-linear

Este trabalho foi julgado adequado para a obtenção do título de Engenheira Civil junto à Universidade Federal de Santa Catarina.

Banca Examinadora:



Prof. Dr. Rafael Holdorf Lopez - Orientador
Universidade Federal de Santa Catarina

Prof. Dr. Leandro Fleck Fadel Miguel
Universidade Federal de Santa Catarina

Eng. Felipe Carraro

Dedico este trabalho a todos que de alguma forma contibuíram para a minha trajetória na graduação.

AGRADECIMENTOS

Aos meus pais, Mônica e Kazuhito, pelo incentivo aos estudos e por toda a dedicação ao longo da minha vida.

Ao meu orientador Rafael, por todos os ensinamentos compartilhados e pela disponibilidade de esclarecer minhas dúvidas.

A todos os professores, por transmitirem com qualidade e generosidade seus conhecimentos e experiências.

Aos amigos que estiveram presentes ao longo da minha graduação, pela companhia nos momentos de alegria e nos momentos difíceis.

RESUMO

O presente trabalho abordou os conceitos de otimização de forma prática, aplicando-os a estruturas de pórticos planos metálicos. Foi utilizado o algoritmo *SGA* (*Search Group Algorithm*) e análise não-linear, aplicando-se em três problemas de pórticos planos já estudados na literatura. Os resultados obtidos foram comparados com o estudo desenvolvido por Carraro (2015), onde foi utilizado o mesmo algoritmo, porém com análises lineares e a consideração dos efeitos de segunda ordem pelo método da amplificação de esforços solicitantes. O número de avaliações da função objetivo para os dois primeiros problemas foi o mesmo, sendo que no terceiro problema fez-se necessário um maior número de avaliações. Neste estudo, todos os problemas apresentaram valores de ótimo global maiores em função das diferenças nos esforços obtidos pelos dois métodos de análise. Desta maneira, foi possível verificar que as soluções para os problemas de pórtico plano estudados encontradas com o uso do método da amplificação de esforços solicitantes resultariam em estruturas falhas, justificando o uso de métodos de análise de segunda ordem rigorosos.

Palavras-chave: Otimização estrutural. Pórtico. SGA. Análise não-linear. Aço.

ABSTRACT

This document discusses the optimization concepts in a practical way, applying them to planar frame steel structures. It was used the *SGA* algorithm (*Search Group Algorithm*) and non-linear analysis, applying it on three problems of planar frames already studied in the literature. The results were compared to the study developed by Carraro (2015), using the same algorithm, but with linear analysis following second-order effects considerations of the AISC-LRFD specification. It was used the same number of evaluations for the first two problems, and in the third problem it was necessary a larger number of evaluations for the objective function. In this study, all problems resulted in higher global optimum values due to the different forces obtained by the two methods of analysis. Therefore, the solutions obtained for the planar frame problems using linear analysis resulted in unsafe structures, justifying the use of non-linear analysis.

Keywords: Structural optimization. Frame. SGA. Non-linear analysis. Steel.

LISTA DE FIGURAS

Figura 1	Mínimos locais e globais.	22
Figura 2	Fluxograma do algoritmo <i>SGA</i>	32
Figura 3	Problema exemplo.	38
Figura 4	Momentos fletores - Análise não-linear - SAP2000.	39
Figura 5	Momentos fletores - Análise não-linear - MASTAN2.	40
Figura 6	Esforços axiais - Análise não-linear - SAP2000.	41
Figura 7	Esforços axiais - Análise não-linear - MASTAN2.	42
Figura 8	Deslocamentos - Análise não-linear - SAP2000.	43
Figura 9	Deslocamentos - Análise não-linear - MASTAN2.	44
Figura 10	Seção do perfil I.	47
Figura 11	Problema 1: Pórtico plano de 2 vãos e 3 pavimentos. ..	53
Figura 12	Curvas de convergência - Problema 1.	56
Figura 13	Diversidade da população - Problema 1.	57
Figura 14	Restrição por elemento - Problema 1 - Pezeshk e Chen (2000).	58
Figura 15	Restrição por elemento - Problema 1 - <i>SGA</i> (Carraro, 2015).	58
Figura 16	Restrição por elemento - Problema 1 - <i>SGA</i> (Este estudo).	59
Figura 17	Problema 1: Pórtico plano de 1 vão e 10 pavimentos. ..	66
Figura 18	Curvas de convergência - Problema 2.	68
Figura 19	Diversidade da população - Problema 2.	68
Figura 20	Restrição por elemento - Problema 2 - <i>SGA</i> (Carraro, 2015).	69
Figura 21	Restrição por elemento - Problema 2 - <i>SGA</i> (Este estudo).	69
Figura 22	Problema 1: Pórtico plano de 3 vãos e 24 pavimentos. .	76
Figura 23	Curvas de convergência - Problema 3.	78
Figura 24	Diversidade da população - Problema 3.	79
Figura 25	Restrições - Problema 3 - <i>SGA</i> (Carraro, 2015).	79
Figura 26	Restrições - Problema 3 - <i>SGA</i> (Este estudo).	80

LISTA DE TABELAS

Tabela 1	Deslocamentos - Análise não-linear - SAP2000.	43
Tabela 2	Deslocamentos - Análise não-linear - MASTAN2.	44
Tabela 3	Relação largura/espessura para elementos sujeitos à flexão.	49
Tabela 4	Resultados para o pórtico plano de 2 vãos e 3 pavimentos.	55
Tabela 5	Resumo das execuções do algoritmo.	55
Tabela 6	Resultados do Problema 1 obtidos por Pezeshk e Chen (2000).	57
Tabela 7	Esforços e restrições - Método da amplificação - Problema 1.	60
Tabela 8	Deslocamentos - Método da amplificação - Problema 1.	61
Tabela 9	Esforços e restrições - Análise não-linear - Problema 1..	62
Tabela 10	Deslocamentos - Análise não-linear - Problema 1.	63
Tabela 11	Resultados para o pórtico plano de 1 vão e 10 pavimentos.	67
Tabela 12	Resumo das execuções do algoritmo.	67
Tabela 13	Esforços e restrições - Método da amplificação - Problema 2.	70
Tabela 14	Deslocamentos - Método da amplificação - Problema 2.	71
Tabela 15	Esforços e restrições - Análise não-linear - Problema 2..	72
Tabela 16	Deslocamentos - Análise não-linear - Problema 2.	73
Tabela 17	Resultados para o pórtico plano de 3 vãos e 24 pavimentos.	77
Tabela 18	Resumo das execuções do algoritmo.	78

SUMÁRIO

1	INTRODUÇÃO	19
1.1	CONSIDERAÇÕES INICIAIS	19
1.2	OBJETIVOS	20
1.2.1	Objetivo Geral	20
1.2.2	Objetivos Específicos	20
2	OTIMIZAÇÃO	21
2.1	HISTÓRICO DA OTIMIZAÇÃO ESTRUTURAL	21
2.2	ALGORITMOS DE OTIMIZAÇÃO	22
2.3	PROBLEMAS DE OTIMIZAÇÃO	23
2.4	FORMULAÇÃO DO PROBLEMA	25
3	SEARCH GROUP ALGORITHM	27
3.1	INTRODUÇÃO	27
3.2	POPULAÇÃO INICIAL	28
3.3	GERAÇÃO DO GRUPO DE BUSCA	28
3.4	MUTAÇÃO DO GRUPO DE BUSCA	28
3.5	PROCESSO ITERATIVO GLOBAL	29
3.6	PROCESSO ITERATIVO LOCAL	30
3.7	RESUMO DO PROCESSO	32
4	ANÁLISE ESTRUTURAL	33
4.1	ANÁLISE DE SEGUNDA ORDEM	33
4.1.1	Imperfeições Iniciais	35
4.1.2	Solução incremental-iterativa	35
4.1.3	Não-linearidade física	37
4.2	COMPARAÇÃO DOS RESULTADOS OBTIDOS PELO MASTAN2 E PELO SAP2000	38
5	DIMENSIONAMENTO	45
5.1	RESISTÊNCIA A ESFORÇOS AXIAIS DE TRAÇÃO ...	45
5.2	RESISTÊNCIA A ESFORÇOS AXIAIS DE COMPRESSÃO	46
5.3	RESISTÊNCIA À FLEXÃO	48
5.3.1	Plastificação da seção transversal	50
5.3.2	Flambagem lateral com torção	50
6	PROBLEMAS ESTUDADOS	53
6.1	PÓRTICO PLANO DE 2 VÃOS E 3 PAVIMENTOS ...	53
6.1.1	Problema	53
6.1.2	Resultados	55
6.1.3	Discussões	64

6.2	PÓRTICO PLANO DE 1 VÃO E 10 PAVIMENTOS	65
6.2.1	Problema	65
6.2.2	Resultados	67
6.2.3	Discussões	74
6.3	PÓRTICO PLANO DE 3 VÃOS E 24 PAVIMENTOS . . .	75
6.3.1	Problema	75
6.3.2	Resultados	77
6.3.3	Discussões	81
7	CONCLUSÃO	83
8	REFERÊNCIAS	85
	APÊNDICE A – Rotinas computacionais - Função objetivo	91
	APÊNDICE B – Rotina computacional - SGA . . .	111

1 INTRODUÇÃO

1.1 CONSIDERAÇÕES INICIAIS

É um desafio para os engenheiros projetar estruturas eficientes e de baixo custo sem comprometer a integridade do sistema. O sucesso da escolha do projeto depende da intuição, experiência e habilidade que o engenheiro possui durante as etapas de lançamento da estrutura, na escolha dos materiais, no dimensionamento das peças. Entretanto, em se tratando de estruturas mais detalhadas que estejam sujeitas a restrições e carregamentos complexos, o projetista encontrará dificuldades na escolha da estrutura que apresente o melhor desempenho, podendo optar por soluções antieconômicas, além de consumir muito tempo durante esse processo.

O desenvolvimento das ferramentas computacionais possibilitou uma evolução nos métodos de solução dos problemas de engenharia. Através do uso dos métodos de otimização, é possível adicionar dados ao projeto que vão além da experiência do projetista, encontrando sistemas estruturais mais vantajosos e que atendam às restrições de projeto, tudo isso em um curto espaço de tempo.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Alunos da graduação de Engenharia Civil, como Carlon (2013), Barbaresco (2014) e Souza (2014), desenvolveram trabalhos que tratavam da otimização geral ou de estruturas treliçadas. Carraro (2015) deu sequência aos estudos ao aplicar conceitos de otimização a pórticos planos de aço.

No sentido de dar continuidade a esses estudos, o objetivo deste trabalho consiste em aplicar o método de otimização “algoritmo do grupo de busca” no projeto ótimo de pórticos planos de aço, utilizando como base o algoritmo desenvolvido por Gonçalves (2015) e adaptado por Carraro (2015) para problemas de pórtico, de forma a considerar uma análise não-linear da estrutura.

1.2.2 Objetivos Específicos

De modo a atingir o objetivo geral, foram estabelecidos os seguintes objetivos específicos:

1. Estudo inicial de conceitos de otimização e das rotinas do algoritmo SGA;
2. Adaptação das rotinas computacionais de análise não-linear ao dimensionamento dos elementos estruturais segundo a norma de aço americana;
3. Aplicação em problemas da literatura;
4. Comparação dos resultados obtidos com os resultados de estudos anteriores.

2 OTIMIZAÇÃO

2.1 HISTÓRICO DA OTIMIZAÇÃO ESTRUTURAL

Segundo SILVA (2001), o primeiro cientista a aplicar o conceito de otimização estrutural foi Maxwell em 1872. Enquanto os engenheiros se preocupavam em desenvolver modelos para calcular com precisão as tensões mecânicas, Maxwell decidiu obter um projeto de ponte que utilizasse a menor quantidade de material e não possuísse risco de falha.

A ideia de Maxwell foi retomada por Michell em 1904, na formulação de um modelo que investigasse a posição dos elementos estruturais de maneira a utilizar o menor volume de material. Não houve desenvolvimento significativo no estudo da otimização estrutural até o advento dos computadores de alta velocidade.

De acordo com Maalawi e Badr (2009), no início da década de 1960 Schimit sugeriu que métodos de aproximação numérica, como o método dos elementos finitos, poderiam ser combinados com técnicas de programação para o desenvolvimento de algoritmos de otimização.

A partir da década de 70, vários algoritmos de otimização para problemas não-lineares foram implementados, onde diferentes abordagens foram apresentadas e aplicadas a uma variedade de problemas de engenharia.

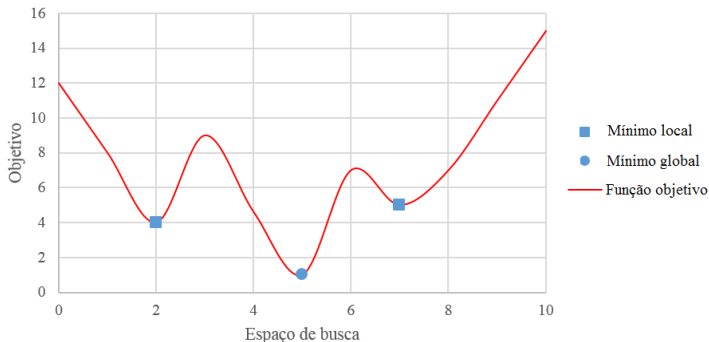
2.2 ALGORITMOS DE OTIMIZAÇÃO

Para auxiliar na solução de problemas de otimização, existe uma variedade de ferramentas disponíveis. O método que resultará no melhor resultado dependerá das características do problema a ser resolvido.

Os algoritmos de otimização clássicos apresentam de modo geral um comportamento determinístico, ou seja, é possível prever todos os seus passos conhecendo seu ponto de partida e sempre levam à mesma resposta ao partir de um mesmo ponto inicial. Os problemas de otimização de engenharia são geralmente complexos, podendo apresentar um comportamento não linear, não-diferenciável. Para esses problemas, a aplicação de métodos de otimização clássicos determinísticos pode não ser satisfatória, justificando assim o uso de técnicas estocásticas.

Algoritmos metaheurísticos apresentam um caráter estocástico, de modo que possuem aleatoriedade em sua formulação. Nestes métodos, várias escolhas são feitas com base em números aleatórios, determinados durante a execução do código. Como a cada execução do código os números escolhidos serão diferentes, um método aleatório não executará a mesma sequência de operações em duas execuções sucessivas, levando a uma resposta final diferente a cada execução. Esta aleatoriedade contribui para dar condições ao algoritmo de explorar o domínio, escapando de mínimos locais e encontrando o ótimo global ou uma boa solução próximo a ele.

Figura 1 – Mínimos locais e globais.



Fonte: Autora

2.3 PROBLEMAS DE OTIMIZAÇÃO

Segundo Arora (2005) a formulação de um problema de otimização é a construção de um modelo matemático bem definido partindo das descrições físicas do problema.

Primeiramente, deve ser definido o objetivo a ser alcançado. Este objetivo depende de certas características do projeto, chamadas variáveis de projeto. A otimização é o processo de definir quais valores essas variáveis devem possuir para que a solução encontrada seja a melhor solução possível, ou seja, a solução ótima. Muitas vezes, as variáveis podem estar limitadas a certos valores, chamados de restrições.

Portanto, os elementos de um problema de otimização são:

- Função objetivo: é a função que associa as variáveis de projeto e precisa ser minimizada ou maximizada, dependendo das exigências de projeto. Em problemas de otimização de estruturas, o principal objetivo é minimizar o custo ou peso da estrutura.
- Variáveis do projeto: são as incógnitas do problema de otimização, ou seja, os parâmetros que podem ser modificados a fim de melhorar o desempenho da estrutura.
- Restrições: são os critérios de projeto, critérios estabelecidos pela norma ou restrições impostas pelas características dos materiais. São as expressões para as restrições que definem o espaço de busca das variáveis do projeto.

O modelo de um problema de otimização pode ser resumido nas seguintes etapas:

- 1) Encontrar o vetor das variáveis de projeto:

$$\mathbf{x} = (x_1, x_2, \dots, x_n) \quad (2.1)$$

- 2) Para minimizar a função objetivo:

$$f(\mathbf{x}) = f(x_1, x_2, \dots, x_n) \quad (2.2)$$

3) Sujeita às restrições:

$$h_j(\mathbf{x}) = h_j(x_1, x_2, \dots, x_n) = 0; \quad j = 1, \dots, p \quad (2.3)$$

$$g_i(\mathbf{x}) = g_i(x_1, x_2, \dots, x_n) \leq 0; \quad i = 1, \dots, m \quad (2.4)$$

onde p é o número de restrições de igualdade e m é o número de restrições de desigualdade.

2.4 FORMULAÇÃO DO PROBLEMA

O objetivo dos problemas de otimização de pórticos planos estudados neste trabalho é encontrar um conjunto de seções transversais para os elementos de vigas e pilares, de maneira que a estrutura apresente o menor peso possível e atenda às restrições de projeto impostas.

O comprimento das barras dos pórticos será mantido constante, para que os pórticos respeitem imposições da arquitetura, por exemplo. Portanto, o peso final da estrutura ficará em função apenas das dimensões das seções transversais de seus elementos, sendo estas dimensões as variáveis de otimização deste projeto.

Os elementos da estrutura são divididos em grupos que utilizam o mesmo perfil metálico. Para determinar o melhor perfil para cada grupo, o algoritmo varia as seções transversais das barras, a partir de uma lista de perfis disponíveis. Uma vez que os valores das áreas das seções transversais são limitados pelos perfis disponibilizados pela indústria, trata-se de um problema de otimização discreta. O vetor das variáveis de projeto é, portanto, o vetor de índices que indicam o perfil utilizado para cada grupo e elementos:

$$\mathbf{I} = (I_1, I_2, \dots, I_n) \quad (2.5)$$

onde n é o número de grupos.

O vetor de variáveis de projeto deve ser encontrado de modo que seja minimizado o peso da estrutura W , representado pela função objetivo:

$$W = \rho \sum_{i=1}^{N_g} A_i \sum_{j=1}^{N_t} L_j + P \quad (2.6)$$

onde ρ é o peso unitário do material, A_i é a área da seção transversal do perfil adotado para o grupo i , N_t é o número total de membros do grupo i , L_j é comprimento do membro j pertencente ao grupo i e a parcela P refere-se ao tratamento das restrições do problema.

As restrições dos problemas estudados neste trabalho são as limitações da norma americana de estruturas de aço AISC (2010) para que os componentes tenham resistência necessária à flexo-tração ou

flexo-compressão e restrições em relação ao deslocamento máximo da estrutura.

Para lidar com as restrições do problema, fez-se o uso de fatores de penalidade. Técnica conhecida na literatura como “Death Penalty” (Pena de morte). (MEZURA-MONTES, COELLO, 2011). Caso a solução tenha violado alguma restrição, será acrescentado um peso adicional à estrutura, ou seja, uma parcela de penalização P , fazendo com que sejam descartadas as soluções obtidas sem que as restrições sejam atendidas. A parcela de penalização P é definida da seguinte maneira:

$$P = \alpha \sum_{i=1}^{n_r} g_i \quad (2.7)$$

onde α é o fator de penalização, e g_i refere-se a i -ésima restrição.

Para o parâmetro α arbitram-se valores altos o suficiente ($10^9 - 10^{10}$) para excluir qualquer solução que viole uma restrição.

3 SEARCH GROUP ALGORITHM

3.1 INTRODUÇÃO

O *SGA* (*Search Group Algorithm* ou “Algoritmo do grupo de busca”) é um algoritmo metaheurístico de otimização global, capaz de se adequar às particularidades de cada problema por meio da alteração de parâmetros de aleatoriedade e amplitude de busca. Foi desenvolvido pelo aluno Matheus Silva Gonçalves, da Universidade Federal de Santa Catarina. Sua eficácia para a otimização de treliças foi verificada em um recente artigo publicado (GONÇALVES et al., 2015) e para a otimização de pórticos planos no trabalho de Carraro (2015).

Em sua implementação original, o algoritmo avaliava as variáveis de forma contínua. Foram realizadas adaptações por Carraro (2015) nas funções de geração das famílias e da população inicial, para que rotina avaliasse somente variáveis discretas inteiras, que condizem a com a formulação do problema de pórtico.

O *SGA* é composto por cinco etapas: a população inicial, seleção inicial do grupo de otimização, mutação do grupo de otimização, geração das famílias e seleção do novo grupo de otimização. Na primeira etapa uma população inicial é gerada aleatoriamente, em seguida a função é avaliada para cada indivíduo e o grupo de otimização é construído. A cada iteração realizada, o grupo de otimização sofre uma mutação baseada nas estatísticas do grupo atual. Uma vez que o grupo é formado, cada um dos membros gera uma família. Na última etapa um novo grupo de otimização é formado pelo melhor membro de cada família.

3.2 POPULAÇÃO INICIAL

O algoritmo inicia o processo de otimização com a geração de uma população inicial de maneira aleatória em qualquer posição do domínio da função objetivo.

3.3 GERAÇÃO DO GRUPO DE BUSCA

Após a geração da população inicial, a função objetivo de cada indivíduo é avaliada. Um determinado número de indivíduos (n_g) é selecionado e o grupo de busca (*search group*) é formado.

Os primeiros membros a compor o grupo de otimização pertencem ao grupo de elite, que se refere a um número de indivíduos que são selecionados diretamente para o grupo de otimização em função de sua boa colocação ou seu *rank*, ou seja, que apresentam menores valores para a função objetivo.

Definidos os primeiros membros, as vagas remanescentes do grupo de busca são disputadas por meio de um torneio. Nesta etapa um número fixo de indivíduos é escolhido aleatoriamente e realiza-se a avaliação da função objetivo para cada um. Os indivíduos melhor avaliados entram para o grupo de otimização. O tamanho do grupo de otimização é definido como uma porcentagem da população total.

3.4 MUTAÇÃO DO GRUPO DE BUSCA

Após a criação do grupo de busca, ocorre a mutação dos indivíduos pertencentes a esse grupo. A mutação consiste em substituir um número de indivíduos (n_{mut}) por novos indivíduos escolhidos com base em análises estatísticas do grupo de otimização atual. O objetivo desta etapa é introduzir novos membros localizados em regiões distantes dos membros atuais, permitindo a exploração de novas regiões do domínio da função objetivo.

Semelhante ao torneio realizado para a geração do grupo de busca, o processo de escolha do indivíduo a ser substituído é feito através de um torneio inverso. Entretanto, neste caso “vence” o indivíduo que receber a pior avaliação, isto é, aquele que apresentar o pior resultado para a função objetivo terá chances maiores de ser

trocado por um novo membro.

O novo membro originado pela mutação é definido pela seguinte equação:

$$x^{mut} = \bar{x} + t \epsilon \sigma \quad (3.1)$$

onde x^{mut} é o novo membro, \bar{x} e σ são a média e o desvio padrão das coordenadas dos membros do grupo de otimização, ϵ é a variável aleatória, com intervalo de -0,5 a 0,5, e t é o parâmetro que controla a distância do novo indivíduo gerado.

3.5 PROCESSO ITERATIVO GLOBAL

Nesta primeira etapa do processo iterativo, chamada de fase global, o principal objetivo é explorar ao máximo o domínio da função objetivo. Para isto, cada indivíduo do grupo de otimização gera um determinado número de outros indivíduos.

É chamado de família o grupo composto pelo membro do grupo de busca e pelos indivíduos gerados por ele. A família é gerada pela perturbação descrita a seguir:

$$x^{novo} = x + \alpha \epsilon \quad (3.2)$$

onde x^{novo} é o novo membro gerado, x é o indivíduo do grupo de busca e α é o parâmetro de aleatoriedade que controla o tamanho da perturbação.

A cada iteração há uma redução no valor da aleatoriedade α . A atualização desse parâmetro é dada por:

$$\alpha = (\alpha_0 \cdot \beta + \alpha_{min})(l_{sup} - l_{inf}) \quad (3.3)$$

onde α_0 é o valor da aleatoriedade inicial, β é um parâmetro que varia de 0 a 1 e promove a redução da aleatoriedade, α_{min} é o valor de aleatoriedade mínimo e l_{sup} e l_{inf} são os limites superior e inferior da função objetivo, respectivamente.

O fator de redução da aleatoriedade β define a maneira com que o parâmetro α é reduzido a cada iteração. É composto pelo maior valor da ordenada entre duas retas. Este valor decai a após cada iteração completa. Desta forma, são considerados valores de

aleatoriedade iniciais mais altos, permitindo uma maior exploração do domínio, decaindo conforme são conhecidas as regiões mais favoráveis para a localização de mínimos.

O fator de redução da aleatoriedade β é definido pela seguinte equação:

$$\beta = \max \left\{ \begin{array}{l} 1,00 - \frac{4,00k}{n_{iterg}} \\ 0,25 - \frac{0,25k}{n_{iterg}} \end{array} \right. \quad (3.4)$$

onde k é iteração atual do algoritmo e n_{interg} é o numero total de iterações a serem realizadas na fase global.

O número total de indivíduos gerados, que irão compor uma família, é igual à diferença entra a população total e o número de componentes dos grupos de otimização, de modo a manter constante o número de avaliações da função objetivo. Cada família será avaliada separadamente, sendo escolhido para o grupo de busca da próxima iteração o membro mais bem avaliado.

3.6 PROCESSO ITERATIVO LOCAL

Após a exploração de boa parte do domínio durante a fase global, tornam-se conhecidas as regiões mais prováveis para a localização dos mínimos, estas serão exploradas na fase local do processo de otimização.

O processo de otimização local diferencia-se do global na etapa de geração dos novos grupos. Ao passo que na etapa global cada família era analisada individualmente, neste processo, após a geração das famílias, todos os indivíduos são avaliados igualmente para gerar um novo grupo composto pelos membros com melhor avaliação, independente da família que os gerou.

A aleatoriedade também se diferencia para a otimização local. Utiliza-se somente uma reta de decaimento e o fator de redução da aleatoriedade β é definido por:

$$\beta = \frac{n_{iterl} - k}{n_{iterl}} \quad (3.5)$$

onde n_{iterl} é o número de iterações da etapa local e k é iteração

corrente.

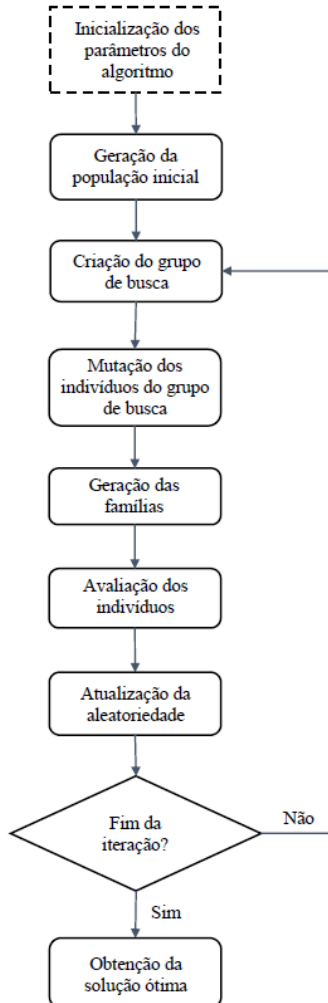
$$\alpha = (\beta\alpha_{min} + r_{min}\alpha_{min})(l_{sup} - l_{inf}) \quad (3.6)$$

onde r_{min} corresponde ao residual de aleatoriedade, definido como uma porcentagem de α_{min} .

O parâmetro α decresce a partir do valor de α_{min} produzido no fim da etapa global, sendo reduzido linearmente com o passar das iterações, acrescido de um pequeno valor residual, de modo a não se obter aleatoriedade nula.

3.7 RESUMO DO PROCESSO

O fluxograma do *SGA* mostrado a seguir sintetiza o funcionamento do algoritmo:

Figura 2 – Fluxograma do algoritmo *SGA*.

4 ANÁLISE ESTRUTURAL

A análise estrutural dos problemas estudados é feita utilizando os códigos do *MASTAN2*, programa similar aos softwares de análise estrutural disponíveis comercialmente de hoje. O *MASTAN2* possibilita realizar análises de primeira ou segunda ordem elástica ou inelástica de pórticos e treliças de duas ou três dimensões sujeitas a cargas estáticas.

Desenvolvido pelos professores de engenharia civil Ronald D. Ziemian e William McGuire da universidade de Bucknell e Cornell respectivamente, o programa permite a entrada gráfica dos dados ou utilizando as rotinas disponibilizadas gratuitamente, além de permitir ao usuário implementar a sua própria análise.

As rotinas de análise estrutural do *MASTAN2* foram implementadas na plataforma *MATLAB*®, um pacote de software para a computação e análise de dados numéricos.

O processo utilizado pelo programa é conhecido como incremental- iterativo e sua análise linear e não-linear baseia-se nos estudos teóricos e formulações numéricas apresentadas no texto *Matrix Structural Analysis, 2nd Edition*, por McGuire, Gallagher, e Ziemian.

4.1 ANÁLISE DE SEGUNDA ORDEM

A análise estrutural depende das características de rigidez e deformabilidade da estrutura, do comportamento das seções, das imperfeições de fabricação e montagem, do comportamento das ligações e, principalmente, da estabilidade dos elementos e da estrutura como um todo.

Segundo a norma de estruturas de aço brasileira ABNT (2008) e a americana AISC (2010), a análise de uma estrutura deve ser feita com um modelo realista, que permita representar a resposta da estrutura e dos materiais estruturais, levando-se em conta as deformações causadas por todos os esforços solicitantes relevantes.

Uma análise não-linear deve ser usada sempre que os deslocamentos afetarem de forma significativa os esforços internos, o que ocorre em pórticos, sendo particularmente importante nos

exemplos estudados neste trabalho, que não possuem contraventamentos impedindo o deslocamento lateral.

A análise em segunda ordem estabelece o equilíbrio da estrutura na posição deformada, gerando esforços adicionais devido à ação das forças aplicadas sobre os deslocamentos. Em pórticos ocorrem dois tipos de efeitos de segunda ordem:

- Efeitos globais ($P-\Delta$): atuantes na estrutura como um todo, são decorrentes dos deslocamentos horizontais dos nós da estrutura.
- Efeitos locais ($P-\delta$): efeitos localizados presente em elementos carregados axialmente, são decorrentes da não-retilinearidade dos eixos das barras.

Os processos de análise de segunda ordem podem ser rigorosos ou simplificados. Nos processos simplificados o equilíbrio é estabelecido na posição deslocada, no entanto, os efeitos não-lineares locais e globais, são introduzidos de forma indireta como, por exemplo, com a aplicação de forças adicionais fictícias ou com a redução da rigidez dos elementos. A resolução de um problema não-linear, neste tipo de análise, é substituída pela resolução de sucessivos problemas lineares de mais fácil solução. Um dos métodos simplificados existentes é o método da amplificação de esforços solicitantes ($B_1 - B_2$).

O método da amplificação de esforços solicitantes consiste em aplicar coeficientes B_1 e B_2 , respectivamente, aos resultados de duas análises lineares a serem superpostas: análise da estrutura impedida de deslocar-se lateralmente e análise da estrutura deslocável sujeita apenas às cargas horizontais iguais às reações obtidas na estrutura. A norma também permite, ao invés de se restringir o deslocamento lateral para o cálculo da estrutura, aplicar somente as cargas verticais da estrutura. Desta forma, a análise de segunda ordem é obtida de forma simplificada.

Nos processo rigorosos é feita, de fato, uma análise não-linear da estrutura com o equilíbrio estabelecido na posição deslocada, introduzindo modificações adequadas na matriz de rigidez da estrutura e resolvendo o problema de forma incremental-iterativa, como será visto a seguir. Também são consideradas imperfeições iniciais através da aplicação de cargas nocionais na estrutura.

Em relação à não-linearidade física do material, o módulo de elasticidade de um material não-linear admitirá diferentes valores para diferentes tensões aplicadas sobre ele. Para a consideração desse efeito, permite-se a adoção de reduções nas rigidezes à flexão e axial.

4.1.1 Imperfeições Iniciais

O efeito de imperfeições iniciais sobre a estabilidade da estrutura deve ser levado em conta na análise da estrutura. Segundo a norma americana AISC (2010), isso pode ser feito pela introdução de cargas nocionais, caracterizadas como forças horizontais equivalentes aplicadas nos nós da estrutura a fim de simular os efeitos desestabilizantes ocasionados pelas imperfeições geométricas iniciais dos elementos.

As cargas nocionais devem ser aplicadas como cargas laterais em todos os pavimentos da estrutura. Devem ser adicionadas a outras cargas laterais no sentido desfavorável. A magnitude dessas cargas é calculada pela seguinte equação:

$$N_i = 0.002Y_i \quad (4.1)$$

onde N_i é a carga nocional aplicada no pavimento i e Y_i são as cargas gravitacionais aplicadas no pavimento i .

Nos problemas estudados neste trabalho, as cargas gravitacionais consideradas foram o peso próprio e as cargas distribuídas aplicadas nas vigas.

4.1.2 Solução incremental-iterativa

Em análises lineares elásticas pelo método dos deslocamentos deve-se resolver a seguinte equação matricial:

$$[K_e]\{\Delta\} = \{P\} \quad (4.2)$$

onde $\{\Delta\}$ é o vetor de deslocamentos dos graus de liberdade da estrutura, $\{P\}$ é o vetor de forças nodais e $[K_e]$ é a matriz de rigidez linear elástica.

Em análises não-lineares, os efeitos da não-linearidade podem ser incorporados por um esquema de solução incremental-iterativa. Os efeitos das deformações e deslocamentos são incluídos na formulação das equações de equilíbrio da seguinte forma:

$$[K_t]\{d\Delta\} = \{dP\} \quad (4.3)$$

onde $\{d\Delta\}$ é o vetor de incremento dos deslocamentos da estrutura, $\{dP\}$ é o vetor de incremento das forças nodais e $[K_t]$ é a matriz de rigidez tangente, que possui duas componentes:

$$[K_t] = [K_e + K_g] \quad (4.4)$$

onde $[K_e]$ é a matriz de rigidez linear elástica e $[K_g]$ é a matriz de rigidez geométrica, que representa as mudanças na rigidez resultantes dos efeitos de segunda ordem.

O processo de solução incremental-iterativa inicia com a obtenção da matriz de rigidez linear elástica $[K_e]$ pelo método dos deslocamentos. Em seguida obtém-se o vetor de incremento das forças nodais $\{dP_i\}$:

$$\{dP_i\} = d\lambda \{P_i\} \quad (4.5)$$

onde $d\lambda$ é o coeficiente que representa o tamanho do incremento, geralmente é um valor definido entre 10 a 20%, e $\{P_i\}$ é o vetor forças nodais da estrutura.

Obtido o vetor de incremento das forças nodais $\{dP_i\}$, o vetor de incremento dos deslocamentos da estrutura $\{d\Delta\}$ é obtido resolvendo o sistema dado pela equação 4.3.

A partir dos vetores de incremento, os valores dos deslocamentos e das forças nodais da estrutura são atualizados e uma nova matriz de rigidez é obtida, dando início à segunda iteração.

Ao final de cada iteração, um critério de convergência é checado. Este critério baseia-se na comparação das deformações obtidas pela iteração atual e pela iteração anterior. Caso a diferença entre elas seja maior que um valor determinado (na ordem de 10^{-2} a 10^{-6}), novas iterações serão realizadas seguindo o mesmo procedimento.

Para lidar com a não-convergência da análise estrutural, foi definido um número máximo de iterações realizadas. Nos problemas de pórtico plano estudados neste trabalho, definiu-se um número máximo de 10 iterações com coeficientes de incremento $d\lambda$ iguais a 0,10.

Ao final da solução incremental-iterativa, o vetor de forças nodais total aplicado na estrutura será: $\{P\} = \sum_{i=1}^n \{dP_i\}$ e o deslocamento correspondente será: $\{\Delta\} = \sum_{i=1}^n \{d\Delta_i\}$, onde n é o número total de incrementos.

4.1.3 Não-linearidade física

A análise da estrutura para determinar as forças resistentes de componentes do pórtico plano deverá usar uma rigidez reduzida, a fim de simular a não-linearidade física.

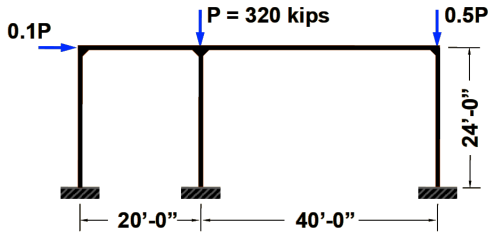
Um fator de 0,80 deve ser aplicado a todas as rigidezes de elementos que contribuem para a estabilidade da estrutura. Sendo permitido também aplicar esse fator de redução para todas as rigidezes na estrutura. Desta forma, para os problemas estudados neste trabalho, foram feitas reduções nas rigidezes à flexão e axial, ou seja, foi utilizado 0,8EI e 0,8EA.

4.2 COMPARAÇÃO DOS RESULTADOS OBTIDOS PELO MASTAN2 E PELO SAP2000

O principal objetivo do problema exemplo que será mostrado a seguir é a comparação dos esforços e deslocamentos obtidos pelo *MASTAN2* com os resultados obtidos com o *SAP2000*, um dos programas de análise estrutural mais utilizados no mundo. O exemplo escolhido foi retirado do tutorial do *MASTAN2* e trata-se de um pórtico plano de dois vãos e um pavimento.

Os elementos são de aço, cujo módulo de elasticidade E é igual a $29000ksi$ ($200GPa$) e a tensão de escoamento F_y é de $36ksi$ ($248,2MPa$). Os pilares possuem seção perfil $W10x45$ e as vigas seção $W27x84$.

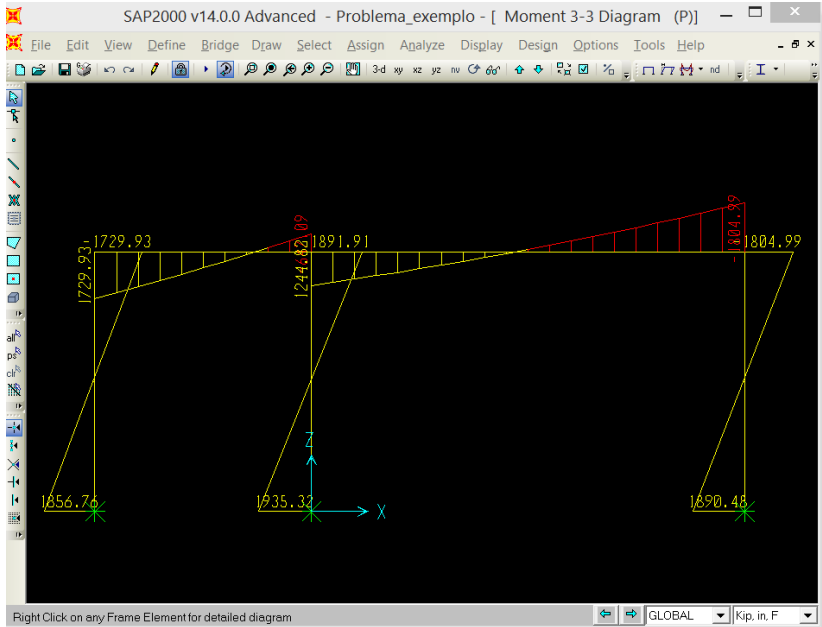
Figura 3 – Problema exemplo.



Fonte: (ZIEMIAN; MCGUIRE, 2007)

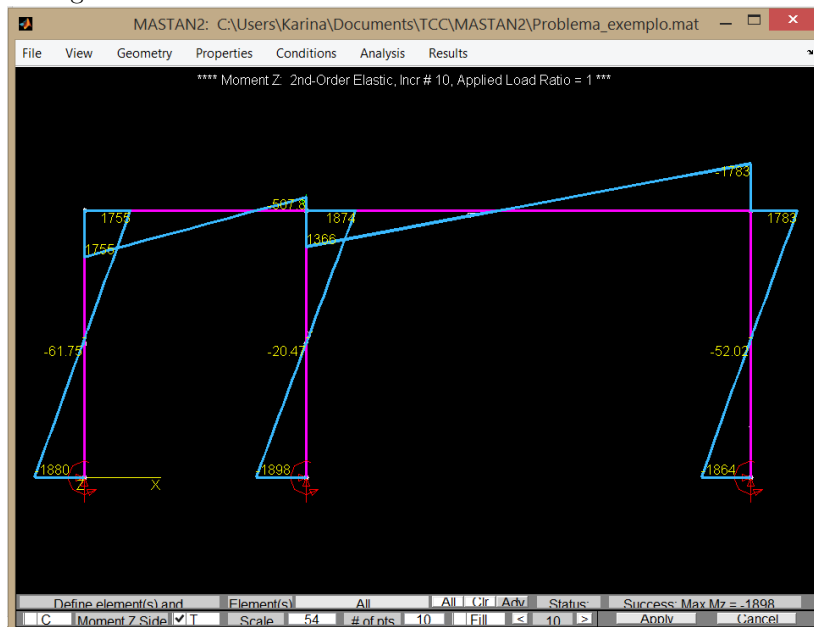
A seguir serão mostrados os diagramas de momentos fletores, esforços axiais e deslocamentos para análises não-lineares obtidos pelos dois programas.

Figura 4 – Momentos fletores - Análise não-linear - SAP2000.



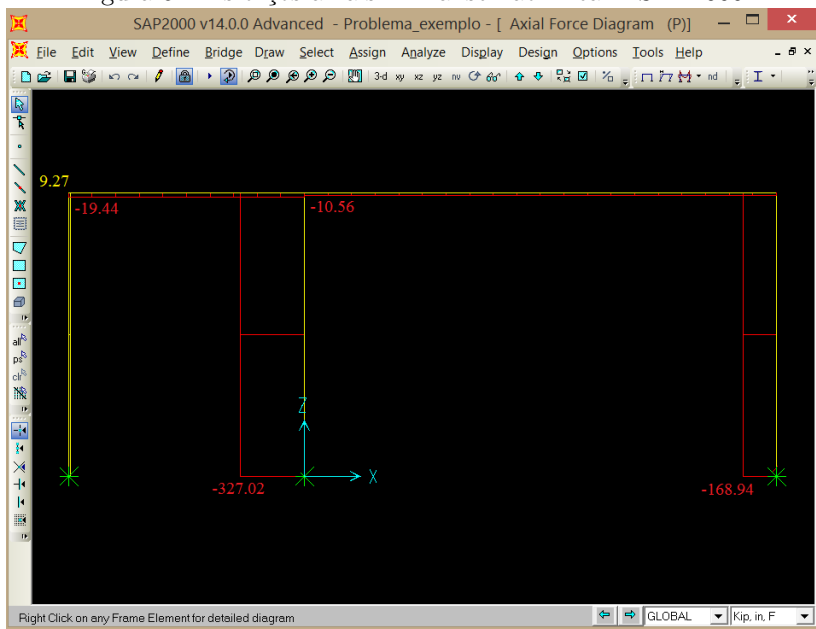
Fonte: (SAP2000 v14.0.0)

Figura 5 – Momentos fletores - Análise não-linear - MASTAN2.



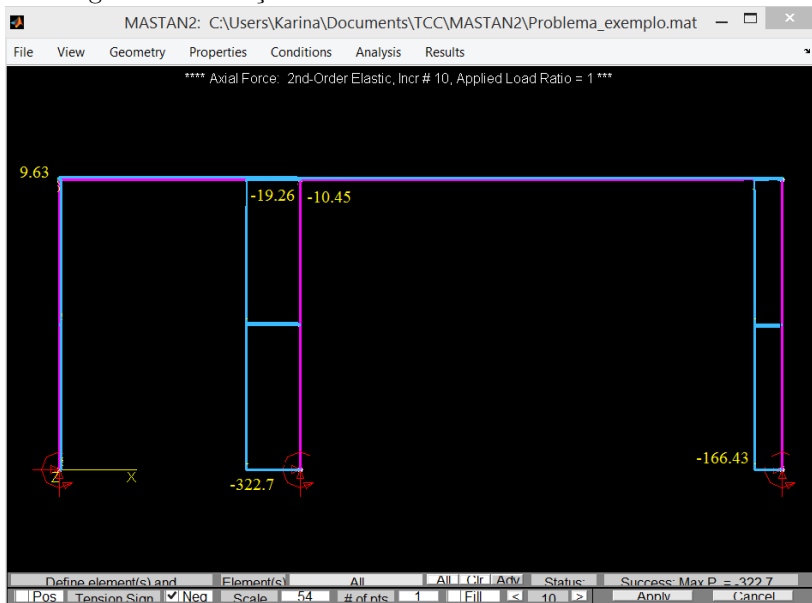
Fonte: (MASTAN2)

Figura 6 – Esforços axiais - Análise não-linear - SAP2000.



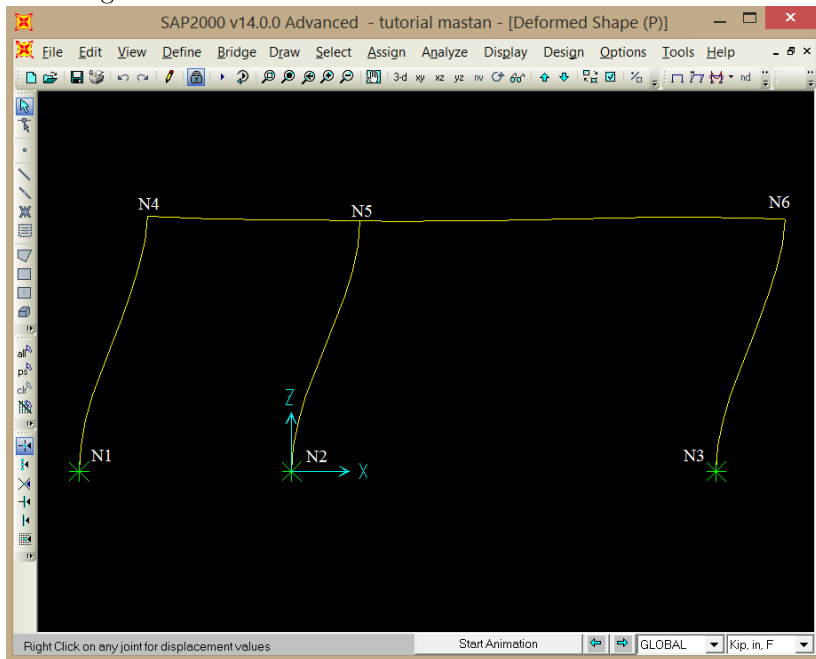
Fonte: (SAP2000 v14.0.0)

Figura 7 – Esforços axiais - Análise não-linear - MASTAN2.



Fonte: (MASTAN2)

Figura 8 – Deslocamentos - Análise não-linear - SAP2000.



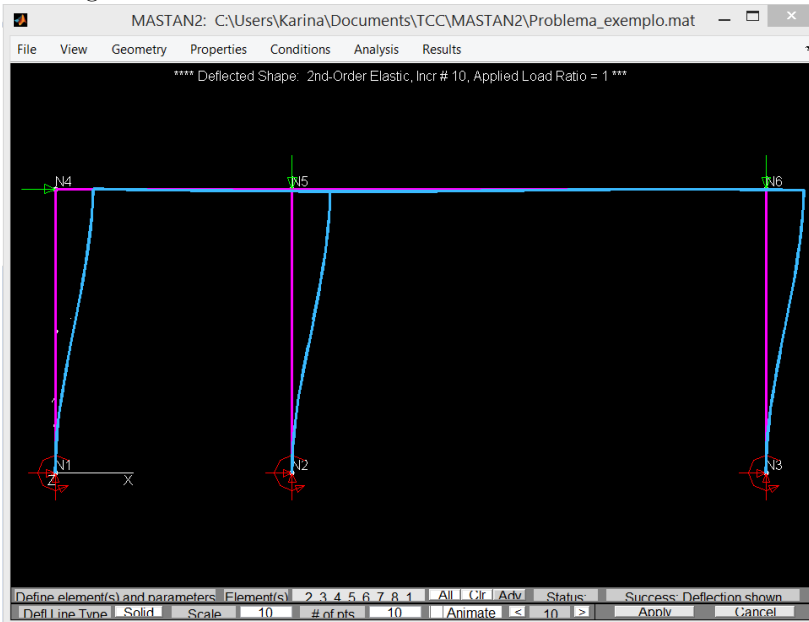
Fonte: (SAP2000 v14.0.0)

Tabela 1 – Deslocamentos - Análise não-linear - SAP2000.

Nó	Deslocamento em x (in)	Deslocamento em y (in)	Rotação em z (rad)
N4	3.902	-0.01991	0.00253
N5	3.895	-0.27014	0.00086
N6	3.888	-0.15200	0.00170

Fonte: Autora

Figura 9 – Deslocamentos - Análise não-linear - MASTAN2.



Fonte: (MASTAN2)

Tabela 2 – Deslocamentos - Análise não-linear - MASTAN2.

Nó	Deslocamento em x (in)	Deslocamento em y (in)	Rotação em z (rad)
N4	3.849	-0.01861	0.002487
N5	3.843	-0.2667	0.000675
N6	3.836	-0.1499	-0.001889

Fonte: Autora

Os resultados obtidos pela entrada gráfica do *MASTAN2* foram os mesmos obtidos utilizando a entrada numérica das rotinas de cálculo.

É possível perceber uma pequena diferença entre alguns dos resultados encontrados para os momentos fletores, esforços axiais e deslocamentos pelos dois programas. Entretanto, os resultados obtidos pelo *MASTAN2* são satisfatórios para a utilização nos problemas que serão estudados neste trabalho.

5 DIMENSIONAMENTO

Os elementos componentes de um pórtico são solicitados por uma atuação simultânea de esforços axiais e momentos fletores. Conforme especificado na norma americana, para que esses componentes tenham a resistência necessária à flexo-tração ou flexo-compressão, deve ser obedecida a limitação fornecida pelas seguintes expressões de interação:

$$\begin{aligned} \frac{P_u}{\phi P_n} + \frac{8}{9} \left(\frac{M_{ux}}{\phi_b M_{nx}} + \frac{M_{uy}}{\phi_b M_{ny}} \right) &\leq 1 \quad \text{se} \quad \frac{P_u}{\phi P_n} \geq 0,2 \\ \frac{P_u}{2\phi P_n} + \left(\frac{M_{ux}}{\phi_b M_{nx}} + \frac{M_{uy}}{\phi_b M_{ny}} \right) &\leq 1 \quad \text{se} \quad \frac{P_u}{\phi P_n} < 0,2 \end{aligned} \quad (5.1)$$

onde P_u é o esforço solicitante axial de tração ou compressão e M_{ux} e M_{uy} são os momentos fletores solicitantes nos eixos x e y. P_n é o esforço resistente axial de tração ou compressão e M_{nx} e M_{ny} são os momentos fletores resistentes nos dois eixos principais da seção. Os multiplicadores ϕ e ϕ_b são os fatores de resistência. O fator ϕ possui o valor de 0,85 para peças comprimidas e 0,90 para peças tracionadas, enquanto que o multiplicador ϕ_b , representa o fator de resistência à flexão e assume o valor de 0,90.

5.1 RESISTÊNCIA A ESFORÇOS AXIAIS DE TRAÇÃO

O esforço axial resistente de tração P_n dos elementos estruturais é obtido de acordo com o estado limite último para o escoamento da seção bruta, calculado pela seguinte equação:

$$P_n = F_y \cdot A_g \quad (5.2)$$

onde F_y é a tensão de escoamento do aço e A_g é a área da seção transversal do perfil.

5.2 RESISTÊNCIA A ESFORÇOS AXIAIS DE COMPRESSÃO

O esforço axial resistente de compressão P_n dos elementos estruturais, sem efeito de flambagem local, é dado pela seguinte equação:

$$P_n = F_{cr} \cdot A_g \quad (5.3)$$

onde F_{cr} é a carga crítica de flambagem e A_g é a área da seção transversal do perfil.

Para elementos não esbeltos sujeitos à compressão, a carga crítica de flambagem F_{cr} é dada por:

$$F_{cr} = \begin{cases} [0,658 \frac{F_y}{F_e}] F_y & \text{se } \frac{KL}{r} \leq 4,71 \sqrt{\frac{E}{F_y}} \\ 0,877 F_y & \text{se } \frac{KL}{r} > 4,71 \sqrt{\frac{E}{F_y}} \end{cases} \quad (5.4)$$

A carga crítica de Euler é obtida pela expressão:

$$F_e = \frac{\pi^2 E}{(\frac{KL}{r})^2} \quad (5.5)$$

Para obter a esbeltez do elemento ($\frac{KL}{r}$) é preciso definir o fator de comprimento efetivo K . Na direção x, que se refere ao deslocamento lateral do pórtico plano, é utilizada a equação proposta por Dumonteil (1992):

$$K = \sqrt{\frac{1,6G_A G_B + 4,0(G_A + G_B) + 7,5}{G_A + G_B + 7,5}} \quad (5.6)$$

onde G_A e G_B são os valores de G nas extremidades da barra analisada, sendo G uma relação entre as rigidezes e comprimentos das vigas e pilares conectados a cada nó, calculado por:

$$G = \frac{\sum(I_{\text{ pilar }}/L_{\text{ pilar }})}{\sum(I_{\text{ viga }}/L_{\text{ viga }})} \quad (5.7)$$

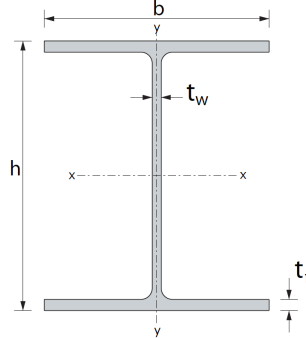
Para nós restringidos, o valor de G é tomado como 10.

Barras submetidas à força axial de compressão, nas quais os elementos componentes da seção transversal são esbeltos, devem adotar um fator de redução Q_a no dimensionamento. A verificação da alma esbelta para perfis do tipo “I” é feita pela seguinte equação:

$$\frac{h}{t_w} < 1,49\sqrt{\frac{E}{F_y}} \quad (5.8)$$

onde h é a altura da alma e t_w sua espessura, como na figura

Figura 10 – Seção do perfil I.



Fonte: (GARDNER, 2012)

Caso o elemento sujeito à compressão possua alma esbelta, a carga crítica de flambagem F_{cr} será calculada por:

$$F_{cr} = \begin{cases} Q_a [0,658 \frac{Q_a F_y}{F_e}] F_y & \text{se } \frac{KL}{r} \leq 4,71\sqrt{\frac{E}{Q_a F_y}} \\ 0,877 F_y & \text{se } \frac{KL}{r} > 4,71\sqrt{\frac{E}{Q_a F_y}} \end{cases} \quad (5.9)$$

O fator de redução Q_a das seções transversais com elementos esbeltos comprimidos é dado por:

$$Q_a = \frac{A_e}{A_g} \quad (5.10)$$

onde A_g é a área da seção transversal e A_e é a área efetiva da

seção transversal, calculada utilizando-se a altura reduzida b_e :

$$b_e = 1,92t_w \sqrt{\frac{E}{F_y}} \left[1 - \frac{0,34}{(h/t_w)} \sqrt{\frac{E}{F_y}} \right] \leq h \quad (5.11)$$

$$A_e = A_g - \sum (h - b_e)t_w \quad (5.12)$$

Elementos metálicos constituídos por chapas finas e sujeitos a esforços de compressão também devem ser verificados para o efeito da flambagem local, um fenômeno de instabilidade de placas comprimidas componentes do perfil. Os perfis de seção “I” utilizados nos problemas foram verificados para flambagem local na alma e na mesa utilizando a seguinte equação:

$$\frac{b}{2t_f} = 0,56 \sqrt{\frac{E}{F_y}} \quad (5.13)$$

onde b é a largura da mesa e t_f sua espessura.

Observou-se que para todos os perfis utilizados neste trabalho, constituídos de aço A36 e de seção “I” da série “W”, o fenômeno de instabilidade de placas não ocorre.

5.3 RESISTÊNCIA À FLEXÃO

A resistência à flexão das barras pode ser afetada pela flambagem local da mesa, pela flambagem local da alma e pela flambagem lateral com torção. A flambagem local, como visto anteriormente para o dimensionamento a esforços de compressão, é a perda da estabilidade de placa, a qual reduz o momento resistente da seção. A flambagem lateral com torção ocorre quando a barra perde seu equilíbrio no plano de flexão e passa a apresentar deslocamentos laterais e rotações de torção.

As seções de elementos sujeitos a momento fletor podem ser classificadas conforme a influência da flambagem local sobre a resistência à flexão. A norma brasileira NBR 8800 (2008) e a norma americana estabelecem as relações largura/espessura limites para as seções compactas (λ_p) e semicompactas (λ_r). As seções que não são classificadas como compactas nem semicompactas são consideradas esbeltas.

- Seção compacta ($\lambda \leq \lambda_p$): a seção pode alcançar o momento de plastificação M_{pl} , exibindo suficiente capacidade de rotação inelástica para configurar uma rótula plástica;
- Seção semicompacta ($\lambda_p < \lambda \leq \lambda_r$): a flambagem local ocorre após o desenvolvimento da plastificação parcial, isto é, com um momento maior que M_y , porém sem apresentar significativa rotação;
- Seção esbelta ($\lambda_r < \lambda$): quando a flambagem local ocorre sem que seja atingido o momento de início de plastificação M_y na seção.

Para perfis do tipo “I”, as relações largura/espessura limites para elementos sujeitos à flexão são:

Tabela 3 – Relação largura/espessura para elementos sujeitos à flexão.

Largura/espessura	Limites	
	λ_p	λ_r
$\frac{b}{2t_f}$	$0,38\sqrt{\frac{E}{F_y}}$	$1,0\sqrt{\frac{E}{F_y}}$
$\frac{h}{2t_w}$	$3,76\sqrt{\frac{E}{F_y}}$	$5,70\sqrt{\frac{E}{F_y}}$

Fonte: (AISC, 2010)

Todos os perfis utilizados nos problemas estudados neste trabalho, constituídos de aço A36 e de seção “I” da série “W”, possuem almas e mesas compactas, com exceção do perfil W6x15, de mesa semi-compacta, sendo tratado na rotina de análise como um caso especial.

Para seções “I” com dois eixos de simetria fletidas em relação ao eixo de maior momento de inércia, em que a alma e a mesa são compactas, devem ser verificados: plastificação da seção transversal e flambagem lateral com torção. O momento nominal M_n deve ser o menor valor obtido para as duas verificações.

5.3.1 Plastificação da seção transversal

A resistência nominal à flexão M_n é para o estado limite de plastificação da seção transversal é igual ao momento de plastificação M_p .

$$M_n = M_p = F_y \cdot Z_y \quad (5.14)$$

onde F_y é a tensão de escoamento do aço e Z_y é o módulo plástico da seção.

5.3.2 Flambagem lateral com torção

A instabilidade é caracterizada por um deslocamento lateral acrescido de uma rotação. Para que haja flambagem lateral, é necessário que a mesa possa se deslocar transversalmente e girar em torno de seu eixo longitudinal, peças com contraventamento contínuo não estão sujeitas à flambagem lateral.

O comportamento de um elemento destravado lateralmente pode ser dividido em três regiões, similarmente à flambagem local. O comprimento destravado L_b , ou seja, o comprimento de trecho entre dois pontos de contenção lateral, é comparado com os valores limites L_p e L_r .

- Elemento curto ($L_b \leq L_p$): ocorre a plastificação total da seção sem que ocorra flambagem lateral;
- Elemento intermediário ($L_p < L_b \leq L_r$): a flambagem lateral ocorre simultaneamente ao escoamento de algumas fibras da seção;
- Elemento longo ($L_b > L_r$): ocorre flambagem lateral antes que as fibras mais solicitadas atinjam a tensão de escoamento.

Os valores limites são calculados pelas seguintes expressões:

$$L_p = 1,76r_y \sqrt{\frac{E}{F_y}} \quad (5.15)$$

$$L_r = 1,95 r_{ts} \frac{E}{0,7F_y} \sqrt{\frac{J}{S_x h_0} + \sqrt{\left(\frac{J}{S_x h_0}\right)^2 + 6,76 \left(\frac{0,7F_y}{E}\right)^2}} \quad (5.16)$$

onde

$$r_{ts}^2 = \frac{\sqrt{I_y C_w}}{S_x} \quad (5.17)$$

Portanto, elementos curtos terão seu momento resistente nominal calculado para o estado limite de plastificação da seção transversal, conforme a equação 5.13.

O momento resistente de elementos intermediários será calculado pela interpolação linear entre o momento de plastificação M_p e o momento M_r . Sendo $M_r = (F_y - 0,3F_y)S_x = 0,7F_y S_x$, representando o desconto 30% do valor da tensão de escoamento do aço referente à tensões residuais.

$$M_n = C_b \left[M_p - (M_p - M_r) \left(\frac{L_b - L_p}{L_r - L_p} \right) \right] \leq M_p \quad (5.18)$$

onde C_b é um fator de modificação que leva em conta o efeito favorável do diagrama de momento fletor uniforme ao longo do trecho L_b , calculado por:

$$C_b = \frac{12,5M_{max}}{2,5M_{max} + 3M_A + 4M_B + 3M_C} \leq 3,0 \quad (5.19)$$

O momento resistente nominal M_n para elementos longos será o valor denominado momento crítico, que será calculado pela seguinte expressão:

$$M_n = F_{cr} \cdot S_x \leq M_p \quad (5.20)$$

$$F_{cr} = \frac{C_b \pi^2 E}{\left(\frac{L_b}{r_{ts}}\right)^2} \sqrt{1 + 0,078 \frac{J_c}{S_x h_0} \left(\frac{L_b}{r_{ts}}\right)^2} \quad (5.21)$$

No caso especial do perfil W6x15, que possui alma compacta e

mesa semi-compacta, além das considerações dos dois estados limites já mencionados, também deve ser feita a verificação do estado limite de flambagem da mesa, utilizando a seguinte equação:

$$M_n = M_p - (M_p - Mr) \left(\frac{\lambda - \lambda_p}{\lambda_r - \lambda_p} \right) \quad (5.22)$$

onde λ é a esbeltez da mesa e λ_p e λ_r são os limites indicados na Tabela 3.

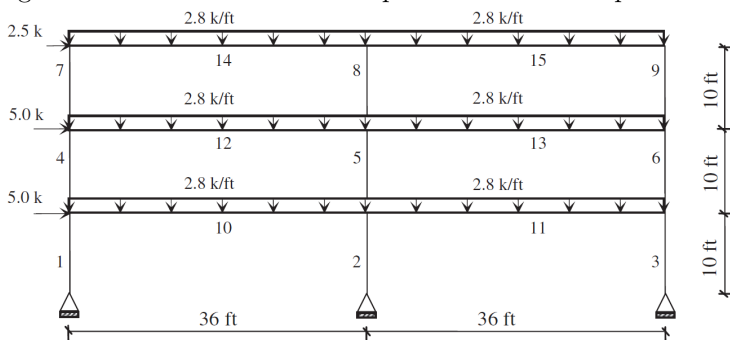
6 PROBLEMAS ESTUDADOS

6.1 PÓRTICO PLANO DE 2 VÃOS E 3 PAVIMENTOS

6.1.1 Problema

O primeiro problema de *benchmark* é um pórtico plano de dois vãos e três pavimentos. Este pórtico já foi otimizado por Pezeshk e Chen (2000), Camp et al. (2005), Degertekin (2008), Togan (2012), Safari e Maheri (2014) e por Carraro (2015) usando o *SGA*.

Figura 11 – Problema 1: Pórtico plano de 2 vãos e 3 pavimentos.



$$1 \text{ k} = 4,45 \text{ kN}$$

$$1 \text{ ft} = 0,305 \text{ m}$$

Fonte: (TOGAN, 2012)

Os elementos são de aço, cujo módulo de elasticidade E é igual a $29000ksi$ ($200GPa$), com uma tensão de escoamento F_y de $36ksi$ ($248,2MPa$) e uma massa específica γ igual a $0,284lb/in^3$ ($7861kg/m^3$). O fator de comprimento efetivo fora do plano K_y foi especificado como 1,0 para os pilares e como um sexto do vão livre para as vigas.

Os valores dos carregamentos uniformes e das cargas nodais já estão fatorados para a aplicação direta das especificações da norma americana (AISC, 2010). Neste problema não foram impostas restrições aos deslocamentos.

As condições de fabricação exigem que todas as seis vigas tenham

perfis de mesma seção, assim como todos os nove pilares devem ter seções idênticas. A divisão em dois grupos torna este um problema de otimização discreta com duas variáveis. A seção do grupo de vigas foi selecionada da lista de perfis do tipo “I”, série “W”, da AISC; entretanto, o grupo de pilares foi limitado às seções da série “W10”, seções com 10 polegadas de altura. Foi utilizado o banco de dados com as propriedades de cada perfil disponível no site oficial do AISC. (STEEL. . . , 2015)

6.1.2 Resultados

Para a solução deste problema foram utilizados como parâmetros de entrada do algoritmo: uma população total $n_{pop} = 20$, um grupo de busca com $n_g = 4$ indivíduos, realizando um número de iterações $it = 20$, sendo destas $it_{global}^{max} = 14$, referente à etapa global e com $n_{perturb} = 4$. Para a aleatoriedade utilizou-se $\alpha_0 = 0,45$ e $\alpha_{min} = 0,01$. O número de avaliações da função objetivo é calculado a partir da expressão $(n_{pop} - n_g) \cdot it$.

A seguir estão apresentados os resultados obtidos por Pezeshk e Chen (2000), utilizando um algoritmo genético (*GA*) e análises não-lineares; os resultados obtidos por Carraro (2015), com o uso do *SGA* e considerações simplificadas dos esforços de segunda ordem; e os resultados obtidos neste estudo, utilizando o *SGA* e análises não-lineares.

Tabela 4 – Resultados para o pórtico plano de 2 vãos e 3 pavimentos.

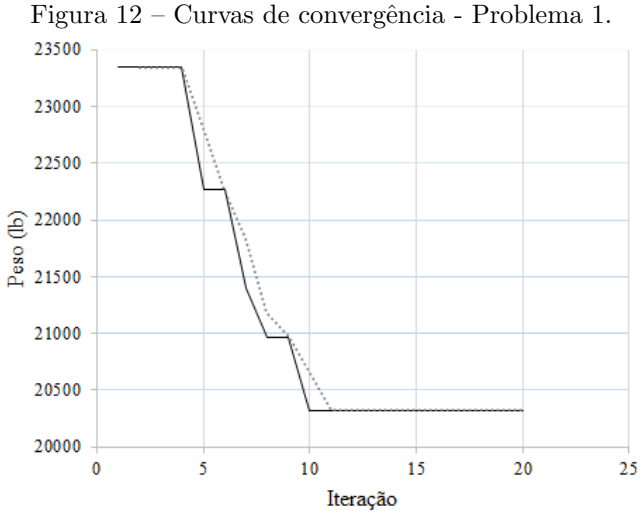
Grupo	Perfis		
	Pezeshk e Chen (2000)	Carraro (2015)	Este estudo
1 (Pilares)	W10x68	W10x60	W10x77
2 (Vigas)	W24x62	W24x62	W24x62
Peso (lb)	19.512	18.792	20.322
Nº de avaliações	1800	320	320

Na Tabela 5 são apresentados os resultados referentes às execuções independentes do algoritmo.

Tabela 5 – Resumo das execuções do algoritmo.

Mínimo observado	20.322 lb
Média	20.736 lb
Desvio padrão	925,73 lb

Na Figura 12, fica representada a convergência do método de busca, ou seja, o melhor valor encontrado para cada iteração.



Fonte: Autora

Outro recurso para visualização do comportamento do algoritmo é a curva “Índice de diversidade x iteração”, onde é possível observar a variação da diversidade da população ao longo da execução do algoritmo. O índice, concebido por Kaveh e Zolghadr (2014), tem a seguinte formulação:

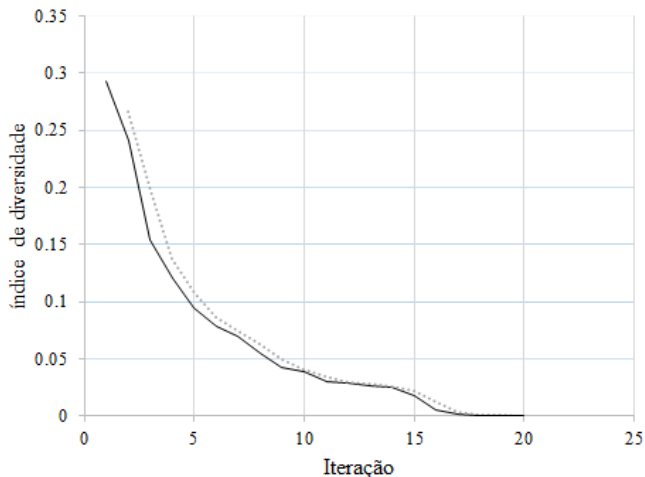
$$\frac{1}{n_{pop}} \sum_{i=1}^{n_{pop}} \sqrt{\sum_{j=1}^{n_g} \left(\frac{O(i) - X_j(i)}{X_{i,max} - X_{i,min}} \right)^2} \quad (6.1)$$

onde $O(i)$ é o valor da i -ésima variável de projeto do melhor indivíduo encontrado até o momento na população e $X_{i,min}$ e $X_{i,max}$ são, respectivamente, os limites superior e inferior de uma dada variável.

Interpreta-se essa formulação da seguinte maneira: quanto menor o índice de diversidade, mais próximos estão os indivíduos.

A curva “Índice de diversidade x iteração” do primeiro problema é mostrada na Figura 13.

Figura 13 – Diversidade da população - Problema 1.



Fonte: Autora

O resultado obtido neste estudo foi comparado com o valor ótimo encontrado por Pezeshk e Chen (2000), autores da literatura que realizaram uma busca exaustiva, ou seja, análises com todas as combinações de perfis possíveis afim de validar os resultados. Seu estudo considerou três casos: no primeiro levou-se em conta uma análise linear ignorando efeitos de segunda ordem especificados na AISC (2010), no segundo caso foi feita uma análise linear considerando efeitos de segunda ordem através de simplificações e no terceiro realizou-se uma análise não-linear.

Tabela 6 – Resultados do Problema 1 obtidos por Pezeshk e Chen (2000).

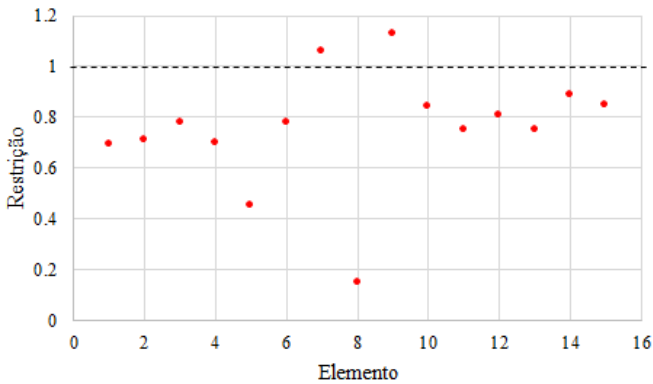
Análise	Pilares	Vigas	Peso (lb)
Caso 1	W24x62	W10x60	18.792
Caso 2	W24x62	W10x60	18.792
Caso 3	W24x62	W10x68	19.512

Os resultados obtidos por Carraro (2015) foram os mesmos obtidos por Pezeshk e Chen (2000) para os dois primeiros casos, que

consideram análises lineares.

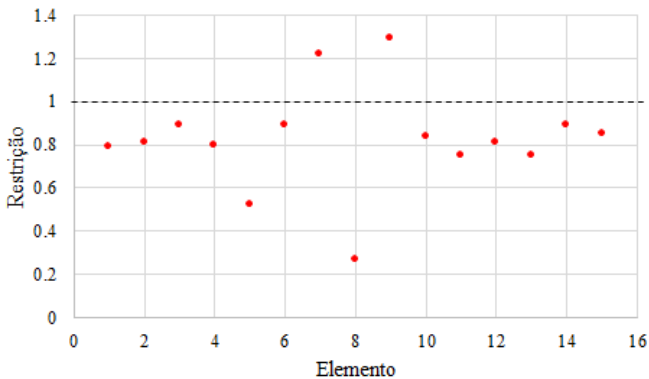
Para avaliar os melhores resultados obtidos pelos autores, estes foram analisados utilizando a função objetivo e as restrições impostas neste trabalho, que se referem às restrições normativas (Equação 5.1). A partir desta avaliação foram elaborados gráficos, indicando o valor da restrição para cada um dos elementos (vigas e pilares). Os gráficos podem ser visualizados nas figuras 15, 15 e 16, a seguir.

Figura 14 – Restrição por elemento - Problema 1 - Pezeshk e Chen (2000).



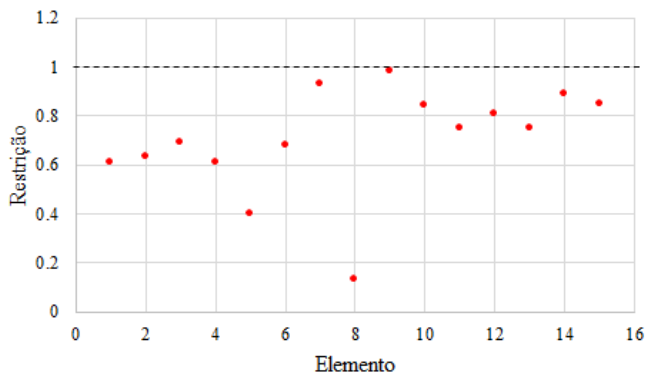
Fonte: Autora

Figura 15 – Restrição por elemento - Problema 1 - SGA (Carraro, 2015).



Fonte: Autora

Figura 16 – Restrição por elemento - Problema 1 - SGA (Este estudo).



Fonte: Autora

Observa-se que ocorreram violações das restrições normativas para as soluções encontradas por Carraro (2015) e por Pezeshk e Chen (2000). Para uma melhor compreensão, será mostrado nas tabelas 7, 8, 9 e 10 os esforços e os deslocamentos obtidos pelo método da amplificação de esforços solicitantes e pela análise não-linear, respectivamente, para a solução encontrada por Carraro (2015), onde os pilares possuem seção W24x62 e as vigas seção W10x60. Junto à tabela dos esforços estão os valores das restrições.

Tabela 7 – Esforços e restrições - Método da amplificação - Problema 1.

Elemento	P_u (kip)	P_r (kip)	M_u (kip)	M_r (kip)	Restrições
1	135.8	564.9	836.5	2685.6	0.57
2	328.9	564.9	325.0	2685.6	0.77
3	140.2	564.9	1220.3	2685.6	0.72
4	90.7	564.9	1283.4	2685.6	0.62
5	219.8	564.9	223.1	2685.6	0.51
6	92.6	564.9	1491.5	2685.6	0.71
7	44.0	564.9	1765.3	2685.6	0.77
8	113.1	564.9	89.3	2685.6	0.26
9	44.5	564.9	1874.3	2685.6	0.82
10	6.1	658.8	4435.8	5544.0	0.89
11	7.5	658.8	3969.4	5544.0	0.80
12	1.4	658.8	4163.1	5544.0	0.84
13	3.6	658.8	3882.3	5544.0	0.78
14	29.1	542.8	4510.5	5544.0	0.93
15	27.9	542.8	4423.6	5544.0	0.92

Fonte: Autora

Tabela 8 – Deslocamentos - Método da amplificação - Problema 1.

Nó	Deslocamento em x (in)	Deslocamento em y (in)	Rotação em z (rad)
1	0	0	0
2	0	0	0
3	0	0	0
4	0.0902	0.0005	-0.0005
5	0.0877	0	-0.0003
6	0.0863	-0.0005	-0.0005
7	0.1698	0.0007	-0.0003
8	0.1669	0	-0.0002
9	0.1657	-0.0007	-0.0003
10	0.2016	0.0008	-0.0001
11	0.2000	0	-0.0001
12	0.1995	-0.0008	-0.0001

Fonte: Autora

Tabela 9 – Esforços e restrições - Análise não-linear - Problema 1.

Elemento	P_u (kip)	P_r (kip)	M_u (kip)	M_r (kip)	Restrições
1	144.53	516.89	1310.6	2671.5	0.80
2	311.29	534.53	461.03	2671.5	0.82
3	148.96	516.89	1559.6	2671.5	0.90
4	96.526	548.92	1701.8	2671.5	0.81
5	207.87	548.92	278.42	2671.5	0.52
6	98.786	548.92	1921.2	2671.5	0.90
7	47.587	548.92	2822.6	2671.5	1.22
8	105.78	548.92	164.28	2671.5	0.27
9	48.269	548.92	3000.4	2671.5	1.30
10	6.0536	658.8	4034.1	5348.7	0.84
11	7.4424	658.8	3591.9	5348.7	0.75
12	9.2536	658.8	3884.3	5348.7	0.81
13	11.287	658.8	3610.2	5348.7	0.76
14	42.815	507.75	4065.6	5348.7	0.89
15	41.062	507.75	3901.4	5348.7	0.86

Fonte: Autora

Tabela 10 – Deslocamentos - Análise não-linear - Problema 1.

Nó	Deslocamento em x (in)	Deslocamento em y (in)	Rotação em z (rad)
1	0	0	0
2	0	0	0
3	0	0	0
4	0.2102	-0.0426	-0.0076
5	0.2166	-0.0917	-0.0019
6	0.2245	-0.0440	0.0031
7	0.4858	-0.0713	-0.0046
8	0.4956	-0.1530	-0.0015
9	0.5076	-0.0733	0.0013
10	0.7052	-0.0854	-0.0112
11	0.6599	-0.1842	-0.0007
12	0.6165	-0.0876	0.0094

Fonte: Autora

6.1.3 Discussões

Um dos objetivos deste primeiro problema foi a possibilidade de realizar diversos testes variando os parâmetros de entrada do algoritmo, por se tratar de um problema de otimização pequeno com apenas duas variáveis. Foi possível observar como os parâmetros influenciam na solução do problema e assim determinar as melhores configurações do algoritmo para os problemas mais complexos que serão vistos a seguir.

Para 320 avaliações da função objetivo, foi encontrado um valor maior para o peso final da estrutura em relação ao estudo feito por Carraro (2015). Este resultado deve-se ao fato de que os esforços obtidos pelo presente estudo, considerando uma análise não-linear do problema, foram maiores em relação aos obtidos por Carraro (2015), que utilizou uma análise linear considerando efeitos de segunda ordem através de simplificações, como observou-se nas tabelas 7, 8, 9 e 10.

O resultado obtido neste estudo também foi maior que o obtido por Pezeshk e Chen (2000). Tal situação pode estar relacionada a diferentes considerações frente a norma, resultando em diferentes avaliações para as restrições do problema.

Em relação ao desempenho do algoritmo, pela Figura 13 nota-se que diversidade se aproxima de zero pelo fato do grupo de busca ser pequeno, com apenas quatro indivíduos e os mesmos apresentarem valores muito próximos.

6.2 PÓRTICO PLANO DE 1 VÃO E 10 PAVIMENTOS

6.2.1 Problema

A otimização de um pórtico de um vão e dez pavimentos é o segundo exemplo de *benchmark* e também já foi estudado por diversos autores da literatura. Este é um exemplo que possui efeitos de segunda ordem mais notáveis.

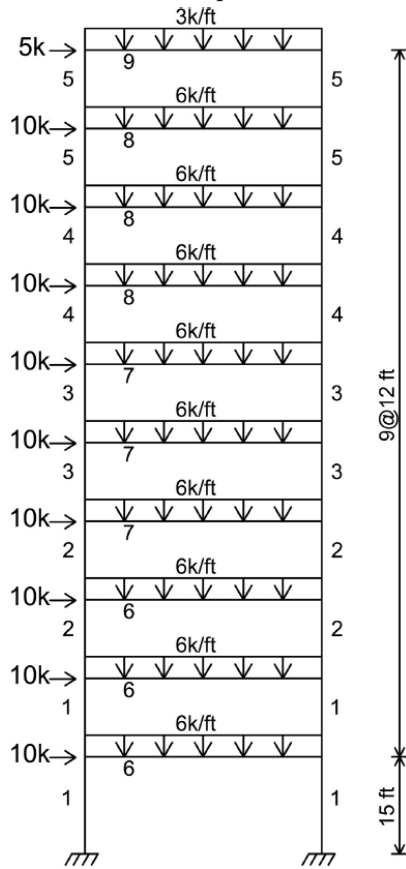
Formado por 30 elementos organizados em 9 grupos, sendo 4 grupos de vigas e 5 de pilares, o problema possui 9 variáveis discretas. A Figura 17 mostra quais membros pertencem a cada grupo de perfis. Para os 4 grupos de vigas podem ser utilizados todos os perfis “I”, da série “W”. Já os grupos de pilares estão restritos ao uso de perfis W12 e W14.

O material é o mesmo do primeiro problema analisado: aço com tensão de escoamento $F_y = 36ksi$ e módulo de elasticidade $E = 29000ksi$.

O problema está sujeito às restrições normativas da norma de projeto de estruturas de aço americana (AISC 2010) e uma restrição de deslocamento, em que o deslocamento entre pavimentos consecutivos não deve ser superior a $h/300$, onde h é a altura do pavimento.

O fator de comprimento efetivo fora do plano K_y foi especificado como 1,0 para os pilares e como 0,2 para as vigas.

Figura 17 – Problema 1: Pórtico plano de 1 vão e 10 pavimentos.



$$1 \text{ k} = 4,45 \text{ kN}$$

$$1 \text{ ft} = 0,305 \text{ m}$$

Fonte: (MAHERI; NARIMANI, 2014)

6.2.2 Resultados

Para a solução deste problema foram utilizados como parâmetros de entrada do algoritmo: uma população total $n_{pop} = 50$, um grupo de busca com $n_g = 10$ indivíduos, realizando um número de iterações $it = 200$, sendo destas $it_{global}^{max} = 140$, referente à etapa global e com $n_{perturb} = 5$. Para a aleatoriedade utilizou-se $\alpha_0 = 0,18$ e $\alpha_{min} = 0,06$.

Na Tabela 11 são apresentados os resultados das análises do Problema 2.

Tabela 11 – Resultados para o pórtico plano de 1 vão e 10 pavimentos.

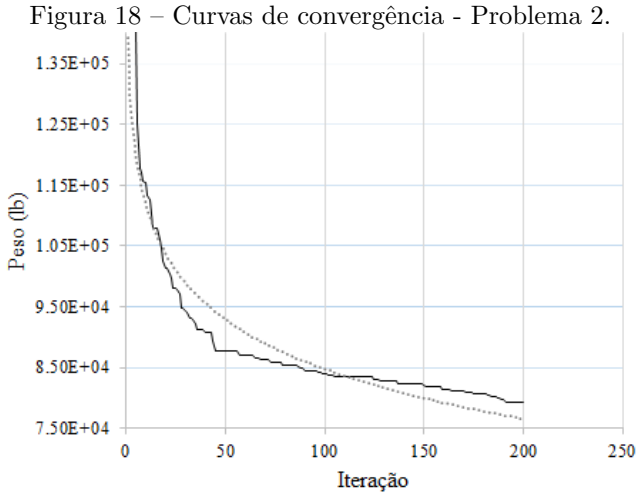
Grupo de elementos	Perfis	
	Carraro (2015)	Este estudo
1 (Pilares Pav. 1-2)	W14x233	W14x257
2 (Pilares Pav. 3-4)	W14x176	W14x283
3 (Pilares Pav. 5-6)	W14x145	W14x159
4 (Pilares Pav. 7-8)	W14x99	W14x109
5 (Pilares Pav. 9-10)	W14x74	W14x132
6 (Vigas Pav. 1-3)	W30x108	W33x118
7 (Vigas Pav. 4-6)	W30x90	W30x108
8 (Vigas Pav. 7-9)	W24x84	W30x90
9 (Vigas Pav. 10)	W24x62	W24x55
Peso (lb)	63.534	76.752
Nº de avaliações	6.000	8.000

Na Tabela 12 são apresentados os resultados referentes às execuções independentes do algoritmo.

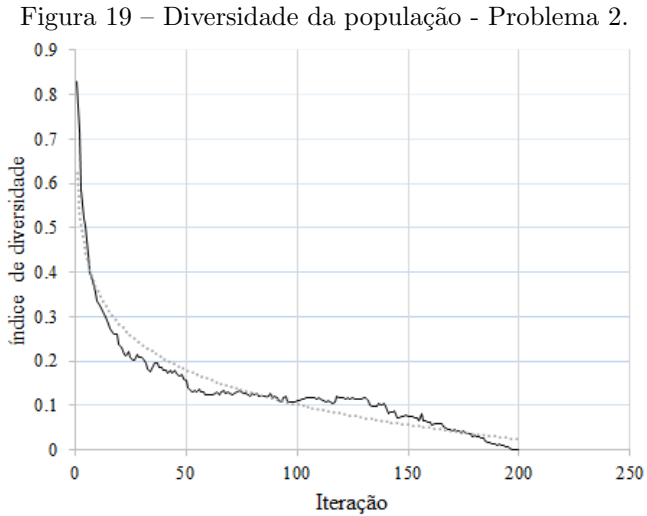
Tabela 12 – Resumo das execuções do algoritmo.

Mínimo observado	76.752 lb
Média	79.200 lb
Desvio padrão	2470.1 lb

Na Figura 18 observa-se a curva de convergência para este problema.



A Figura 19 ilustra a diversidade da população ao longo da execução do método.



Nas Figuras 20 e 21 estão representados os gráficos das restrições associadas a cada elemento, para o melhor valor encontrado por Carraro (2015) e por este estudo, respectivamente.

Figura 20 – Restrição por elemento - Problema 2 - *SGA* (Carraro, 2015).

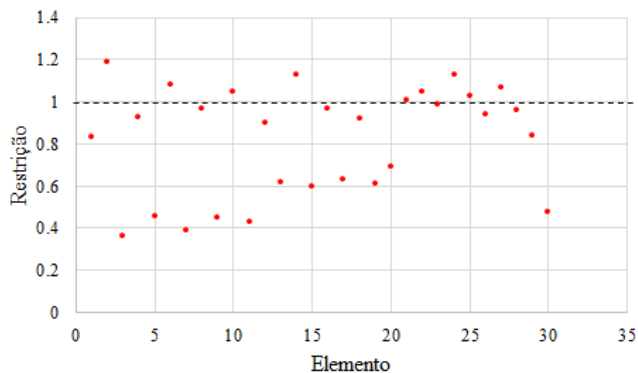
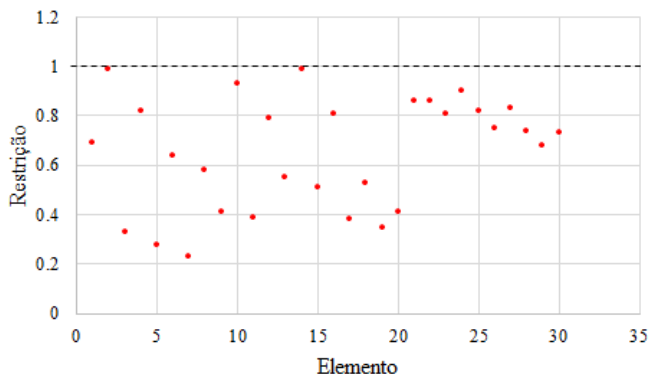


Figura 21 – Restrição por elemento - Problema 2 - *SGA* (Este estudo).



Da mesma forma que para o primeiro problema, o melhor entendimento dos resultados é obtido com a observação dos valores dos esforços e deslocamentos obtidos pelos dois métodos de análise de segunda ordem. A seguir serão mostrados os resultados das análises para o pórtico do problema 2 com as mesmas seções dos elementos obtidas por Carraro (2015), com o uso do método da amplificação e de análises não-lineares, respectivamente.

Tabela 13 – Esforços e restrições - Método da amplificação - Problema 2.

Elemento	P_u (kip)	P_r (kip)	M_u (kip)	M_r (kip)	Restrições
1	665.31	2228.1	5080	15696	0.65
2	1044.7	2228.1	6954.1	15696	0.96
3	607.67	2310.9	608.2	15696	0.33
4	922.33	2310.9	6219.5	15696	0.83
5	551.79	1743	744.5	11520	0.42
6	798.21	1743	5302.3	11520	0.96
7	493.3	1743	101.57	11520	0.32
8	676.7	1743	5058.4	11520	0.87
9	428.55	1434.8	210.73	9360	0.35
10	561.45	1434.8	4544.4	9360	0.91
11	360.6	1434.8	832.75	9360	0.37
12	449.4	1434.8	4434.9	9360	0.82
13	288.39	967.72	917.57	6228	0.48
14	341.61	967.72	3573.7	6228	0.96
15	210.61	967.72	1626.9	6228	0.50
16	239.39	967.72	3570.1	6228	0.84
17	129.19	723.49	1718.3	4909.5	0.49
18	140.81	723.49	2974.9	4909.5	0.81
19	43.568	723.49	2114.9	4909.5	0.51
20	46.432	723.49	2334.8	4909.5	0.56
21	19.096	1141.2	10680	12456	0.96
22	13.81	949.96	10568	12456	0.95
23	0.75497	964.85	9886.3	12456	0.88
24	8.3559	759.07	8983.9	10188	0.99
25	1.4833	766.08	8339.2	10188	0.91
26	10.288	775.92	7626.3	10188	0.84
27	0.24557	724.71	6737.5	8064	0.93
28	8.3929	734.75	6116.4	8064	0.85
29	6.2186	744.93	5281.4	8064	0.73
30	32.342	689.59	2324.7	7200	0.38

Fonte: Autora

Tabela 14 – Deslocamentos - Método da amplificação - Problema 2.

Nó	Deslocamento em x (in)	Deslocamento em y (in)	Rotação em z (rad)
1	0	0	0
2	0	0	0
3	0.5206	0.0161	-0.0028
4	0.5187	-0.0161	-0.0028
5	1.0467	0.0267	-0.0028
6	1.0447	-0.0267	-0.0028
7	1.5844	0.0375	-0.0026
8	1.5825	-0.0375	-0.0026
9	2.0928	0.0457	-0.0027
10	2.0904	-0.0457	-0.0027
11	2.5915	0.0528	-0.0024
12	2.5891	-0.0528	-0.0024
13	3.0209	0.0577	-0.0020
14	3.0186	-0.0577	-0.0020
15	3.4558	0.0619	-0.0021
16	3.4533	-0.0619	-0.0021
17	3.8214	0.0642	-0.0016
18	3.8189	-0.0642	-0.0016
19	4.0845	0.0654	-0.0010
20	4.0821	-0.0654	-0.0010
21	4.2261	0.0657	-0.0006
22	4.2247	-0.0657	-0.0006

Fonte: Autora

Tabela 15 – Esforços e restrições - Análise não-linear - Problema 2.

Elemento	P_u (kip)	P_r (kip)	M_u (kip)	M_r (kip)	Restrições
1	703.8	1910.2	6707.8	15696	0.83
2	1105.3	1910.2	8743.1	15696	1.19
3	642.0	2221.5	753.4	15696	0.37
4	976.0	2221.5	7141.6	15696	0.94
5	582.1	1683.3	902.6	11520	0.46
6	845.1	1683.3	6161.6	11520	1.09
7	519.2	1687.8	568.1	11520	0.39
8	717.3	1687.8	5883.5	11520	0.98
9	451.8	1387.4	843.5	9360	0.45
10	594.0	1387.4	5450.5	9360	1.05
11	379.4	1401.1	1307.0	9360	0.44
12	475.6	1401.1	5008.3	9360	0.91
13	302.2	948.7	1713.8	6228	0.63
14	362.0	948.7	4494.8	6228	1.14
15	220.3	948.4	2159.1	6228	0.60
16	253.1	948.4	4304.5	6228	0.98
17	134.9	692.1	2125.8	5004	0.64
18	149.0	692.1	3482.7	5004	0.93
19	45.1	692.1	2604.1	5004	0.61
20	49.2	692.1	2930.8	5004	0.69
21	19.2	1141.2	11233.0	12456	1.01
22	10.4	891.8	11724.0	12456	1.05
23	3.8	908.8	11156.0	12456	1.00
24	5.1	713.7	10340.0	10188	1.13
25	4.9	721.7	9457.0	10188	1.04
26	5.6	733.0	8580.7	10188	0.94
27	2.8	677.8	7756.6	8064	1.07
28	9.5	688.0	6954.3	8064	0.97
29	3.8	698.2	6102.1	8064	0.84
30	38.5	651.8	2930.8	7200	0.49

Fonte: Autora

Tabela 16 – Deslocamentos - Análise não-linear - Problema 2.

Nó	Deslocamento em x (in)	Deslocamento em y (in)	Rotação em z (rad)
1	0	0	0
2	0	0	0
3	1.044	0.0830	-0.0087
4	1.049	-0.1285	-0.0061
5	2.335	-0.1470	-0.0088
6	2.332	-0.2228	-0.0071
7	3.596	-0.2054	-0.0083
8	3.595	-0.3051	-0.0064
9	4.724	-0.2570	-0.0073
10	4.722	-0.3746	-0.0055
11	5.686	-0.3013	-0.0063
12	5.685	-0.4318	-0.0045
13	6.478	-0.3380	-0.0053
14	6.476	-0.4772	-0.0034
15	7.101	-0.3668	-0.0042
16	7.100	-0.5115	-0.0025
17	7.569	-0.3877	-0.0035
18	7.566	-0.5353	-0.0014
19	7.894	-0.4043	-0.0029
20	7.893	-0.5537	0.0002
21	8.093	-0.4100	-0.0022
22	8.079	-0.5598	0.0003

Fonte: Autora

6.2.3 Discussões

Novamente o algoritmo chegou a um resultado maior em relação ao estudo considerando análises lineares. Nas tabelas 13, 14, 15 e 16 é possível visualizar as diferenças nos esforços axiais, momentos fletores e deslocamentos obtidos.

Nota-se que os elementos que estavam próximos, porém abaixo do limite imposto pelas restrições normativas para a análise pelo método da amplificação, passam a violar essas restrições com a consideração da análise não-linear.

Para este problema, foi possível chegar ao resultado da função objetivo com um número maior de avaliações (8.000) que o utilizado por Carraro (2015). O número máximo de avaliações da função para que o algoritmo conseguisse explorar de forma satisfatória o domínio aumentou em relação ao primeiro problema, visto que o problema anterior continha um espaço de busca de 4806 configurações estruturais, enquanto neste tem-se um espaço com $6,36 \times 10^{18}$ configurações de perfis válidas.

6.3 PÓRTICO PLANO DE 3 VÃOS E 24 PAVIMENTOS

6.3.1 Problema

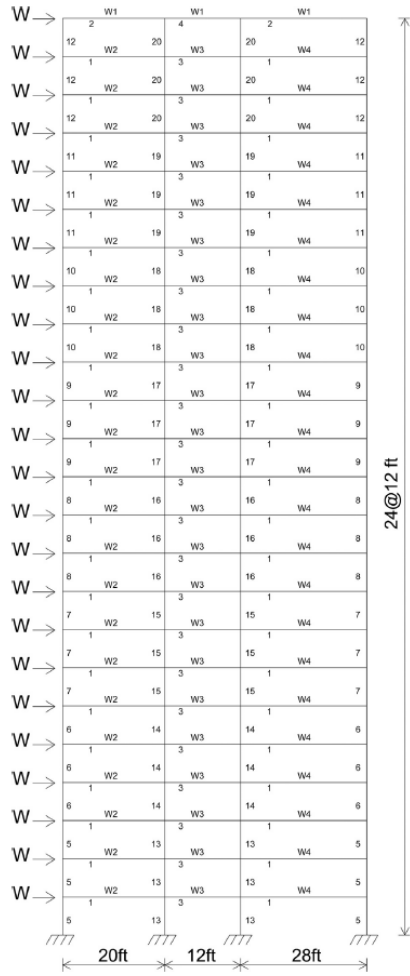
O terceiro exemplo é a otimização de um pórtico plano de três vãos e vinte e quatro pavimentos.

Os 168 membros do pórtico estão divididos em 20 grupos. Os pilares externos estão agrupados a cada três pavimentos, da mesma forma que os pilares internos. As vigas do primeiro e terceiro vão de todos os pavimentos possuem os mesmos perfis e as vigas da cobertura estão separadas em dois grupos diferentes, resultando em quatro grupos de vigas. Este exemplo é, portanto, um problema de otimização discreta com 20 variáveis.

Cada um dos elementos dos quatro grupos de vigas foi escolhido entre os 267 perfis do tipo “I”, série “W”, da AISC (2010), enquanto os 16 grupos de pilares foram limitados às seções da série “W14”.

As propriedades do material se diferenciam dos dois primeiros problemas. O módulo de elasticidade E é igual a $29732ksi$ ($205GPa$), a tensão de escoamento F_y é de $33,4ksi$ ($230,3MPa$) e a massa específica γ igual a $0,284lb/in^3$ ($7861kg/m^3$).

Figura 22 – Problema 1: Pórtico plano de 3 vãos e 24 pavimentos.



Fonte: (MAHERI; NARIMANI, 2014)

6.3.2 Resultados

Para a solução deste problema foram utilizados como parâmetros de entrada do algoritmo: uma população total $n_{pop} = 50$, um grupo de busca com $n_g = 10$ indivíduos, realizando um número de iterações $it = 300$, sendo destas $it_{global}^{max} = 150$, referente à etapa global e com $n_{perturb} = 5$. Para a aleatoriedade utilizou-se $\alpha_0 = 0,14$ e $\alpha_{min} = 0,08$.

Na Tabela 17 são apresentados os resultados de cada grupo de projeto obtidos para o problema 3.

Tabela 17 – Resultados para o pórtico plano de 3 vãos e 24 pavimentos.

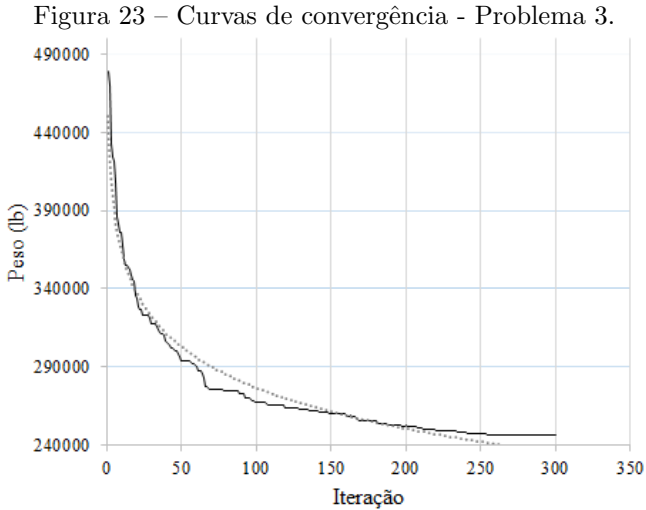
Grupo de elementos	Perfis	
	Carraro (2015)	Este estudo
1	W24x68	W30x90
2	W18x40	W21x48
3	W24x55	W21x68
4	W33x118	W14x26
5	W14x132	W14x145
6	W14x30	W14x159
7	W14x99	W14x159
8	W14x53	W14x74
9	W14x74	W14x99
10	W14x26	W14x61
11	W14x68	W14x43
12	W14x193	W14x68
13	W14x145	W14x211
14	W14x26	W14x159
15	W14x26	W14x90
16	W14x43	W14x74
17	W14x26	W14x120
18	W14x120	W14x159
19	W14x426	W14x48
20	W14x68	W14x82
Peso (lb)	194.400	246.816
Nº de avaliações	8.000	12.000

Na Tabela 18 estão dispostos os resultados para uma série de execuções independentes do algoritmo.

Tabela 18 – Resumo das execuções do algoritmo.

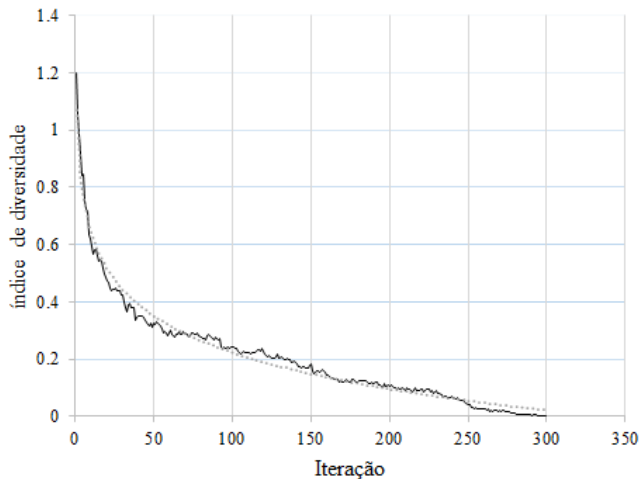
Mínimo observado	246.816 lb
Média	256.550 lb
Desvio padrão	5.876 lb

Na Figura 23 pode ser visualizada a convergência da rotina para este problema.



A diversidade da população ao longo da execução do algoritmo pode ser observada na Figura 24.

Figura 24 – Diversidade da população - Problema 3.



Nas Figuras 25 e 26 podem ser visualizados os gráficos das restrições associadas a cada elemento referentes ao Problema 3.

Figura 25 – Restrições - Problema 3 - SGA (Carraro, 2015).

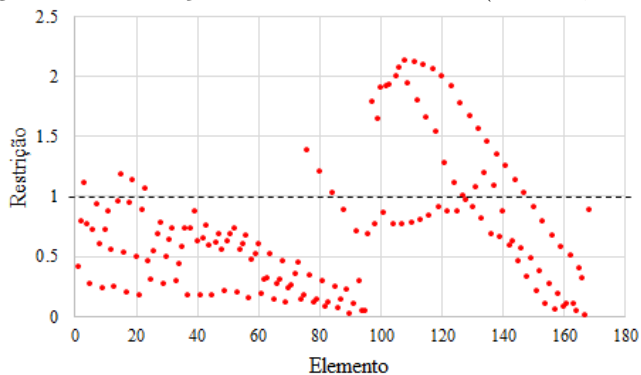
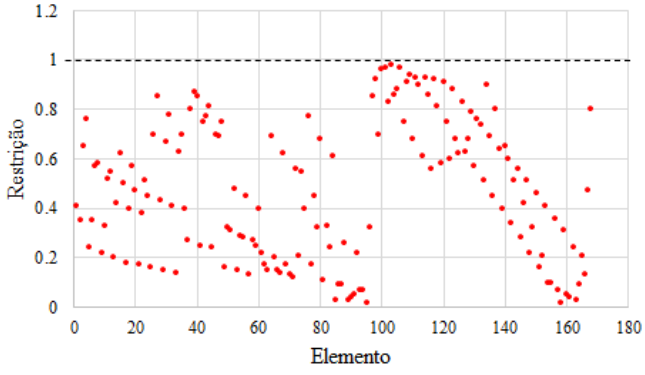


Figura 26 – Restrições - Problema 3 - SGA (Este estudo).



6.3.3 Discussões

Para o terceiro e último problema analisado, obteve-se também um maior peso para a estrutura, para um número de avaliações de 12.000, maior que o utilizado por Carraro (2015), de 8.000 avaliações. Porém, pode-se visualizar na Figura 25 que as restrições ultrapassaram o limite imposto para alguns elementos de viga na solução encontrada por Carraro (2015), sendo que esse limite chegou a ser ultrapassado em mais de duas vezes para os piores casos.

Em relação ao desempenho do algoritmo, verifica-se que a curva de convergência apresenta o formato típico de uma curva de otimização, onde existe uma queda maior do objetivo nas iterações iniciais, havendo ainda um decréscimo mais suave ao longo do resto do processo.

7 CONCLUSÃO

De acordo com os objetivos inicialmente propostos, em linhas gerais, o presente trabalho permitiu uma comparação dos resultados da otimização estrutural do algoritmo *SGA* de problemas de pórticos planos com o uso de análises não-lineares aos resultados obtidos pelo estudo de Carraro (2015), onde foi utilizado o método da amplificação dos esforços para a consideração de efeitos de segunda ordem.

A realização deste estudo foi possível a partir da adaptação dos códigos do *SGA* para pórticos aos códigos de análise estrutural do *MASTAN2*.

Como resultado, obteve-se para os três problemas estudados um maior valor ótimo global comparado às soluções obtidas por Carraro (2015). Os resultados obtidos pelos estudos anteriores, entretanto, não respeitam as restrições normativas, resultando em estruturas falhas.

Nesse sentido, salienta-se a importância de uma análise estrutural não-linear, pois esta representa um modelo realista, que permite a obtenção da resposta da estrutura e dos materiais estruturais, levando-se em conta as deformações causadas por todos os esforços solicitantes relevantes de maneira exata.

Para futuros trabalhos, pode-se realizar uma avaliação mais aprofundada dos resultados obtidos pelo método da análise não-linear e pelo método de análise linear com a consideração dos efeitos de segunda ordem através da amplificação dos esforços. Em relação à continuidade dos estudos sobre otimização estrutural, recomenda-se avaliar o desempenho do algoritmo *SGA* em estruturas mais complexas, como pórticos espaciais. Poderia também ser feita uma avaliação de novas estratégias para lidar com as restrições.

8 REFERÊNCIAS

AMERICAN INSTITUTE OF STEEL CONSTRUCTION. ANSI/AISC 360-10: Specification for structural steel buildings. Chicago, 2010.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. NBR 8800: Projeto de estruturas de aço e de estruturas mistas de aço e concreto de edifícios - procedimento. Rio de Janeiro, 2008.

BARBARESCO, G. M. Otimização de problemas de engenharia utilizando o algoritmo da Competição Imperialista (ICA). 64 f. Monografia (Trabalho de conclusão de curso) - Departamento de Engenharia Civil, Universidade Federal de Santa Catarina, Florianópolis, 2014.

CAMP, C. V.; BICHON, B. J.; STOVALL, S. P. Design of Steel Frames Using Ant Colony Optimization. *Journal of Structural Engineering*, v. 131, n. 3, p. 369-379, 2005. ISSN 0733-9445.

CARLON, A. G. Otimização em treliças de estruturas metálicas aplicando o algoritmo ICA. 74 f. Monografia (Trabalho de conclusão de curso) - Departamento de Engenharia Civil, Universidade Federal de Santa Catarina, Florianópolis, 2013.

CARRARO, F. Otimização estrutural de pórticos planos utilizando o algoritmo SGA. Monografia (Trabalho de conclusão de curso) - Departamento de Engenharia Civil, Universidade Federal de Santa Catarina, Florianópolis, 2015.

DEGERTEKIN, S. Optimum design of steel frames using harmony search algorithm. *Structural and Multidisciplinary Optimization*, Springer-Verlag, v. 36, n. 4, p. 393-401, 2008. ISSN 1615-147X.

DOGAN, E.; SAKA, M. Optimum design of unbraced steel frames to LRFD-AISC using particle swarm optimization. *Advances in Engineering Software*, v. 46, n. 1, p. 27-34, 2012. ISSN 09659978.

DUMONTEIL, P. Simple equations for effective length factors. *Engineering Journal*, v. 29, n. 3, p. 111-115, 1992.

GALAMBOS, T.; SUROVEK, A. *Structural Stability of Steel: Concepts and Applications for Structural Engineers*. [S.l.]: Wiley, 2008. ISBN 9780470037782.

GARDNER, L. Stability of Steel Beams and Columns: In Accordance with Eurocodes and the UK National Annexes. Berkshire, UK: Beliveau Editeur, 2011. ISBN 9781859421994.

GONÇALVES, M. S.; LOPEZ, R. H.; MIGUEL, L. F. F. Search group algorithm: A new metaheuristic method for the optimization of truss structures. *Computers and Structures*, v. 153, p. 165-184, 2015.

MAALAWI, H. Y.; BADR, M. A. Optimization of Mechanical Elements and Structures: a Review with Application. *Journal of Applied Sciences Research.*: 221-231, 2009. INSInet Publication Design.

MAHERI, M. R.; NARIMANI, M. An enhanced harmony search algorithm for optimum design of side sway steel frames. *Computers and Structures*, Elsevier Ltd, v. 136, p. 78-89, 2014. ISSN 00457949. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0045794914000558>>.

MCGUIRE, W.; ZIEMIAN, R. D. Tutorial for MASTAN2. Version 3.0. Berkeley, 2008.

MCGUIRE, W.; GALLAGHER, R. H.; ZIEMIAN, R. D., Matrix Structural Analysis. 2nd Edition. Faculty Books, 2000. ISBN 9781507585139.

MEZURA-MONTES, E.; COELLO, C. A. C. Constraint-handling in natureinspired numerical optimization: Past, present and future. *Swarm and Evolutionary Computation*, v. 1, n. 4, p. 173 - 194, 2011. ISSN 2210-6502. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S2210650211000538>>.

MICHELL, A.G.M.. The Limit of Economy of Material in Framed Structures. *Philosophical Magazine, Series, P.* 589 - 597, 1904.

PEZESHK, S.; CHEN, D. Design of nonlinear framed structures using genetic optimization. *Journal of Structural Engineering*, n. March, p. 382-388, 2000.

PFEIL, W.; PFEIL, M. Estruturas de aço : dimensionamento prático. 8. ed. Rio de Janeiro: LTC, 2014.

SILVESTRE, N.; CAMOTIM, D. Elastic buckling and second-order behaviour of pitched-roof steel frames. *Journal of Constructional Steel Research*, v. 63, n. 6, p. 804-818, 2007. ISSN 0143-974X.

SILVA.E.C.N.. Técnicas de otimização aplicadas no projeto de peças mecânicas. Dissertação - Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos, Escola Politécnica da USP, São Paulo, 2001.

SOUZA, R. R.. Metodologia para otimização topológica, dimensional e geométrica de estruturas de torres de linhas de transmissão. Monografia (Trabalho de conclusão de curso) - Departamento de Engenharia Civil, Universidade Federal de Santa Catarina, Florianópolis, 2014.

STEEL Construction Manual Shapes Database. 2015. Disponível em: <<http://www.aisc.org/content.aspx-id=2868>>. Acesso em: 15 de agosto de 2015.

TOGAN, V. Design of planar steel frames using Teaching-Learning Based Optimization. Engineering Structures, 2012. ISSN 01410296.

APÊNDICE A – Rotinas computacionais - Função objetivo

A.1 ENTRADA DE DADOS

```

1 function [Pu,Mu,Cbs,secoes,vetorKx,n_pilares,restricao_desloc] =
    Portico.nt.lt(perfis)
2 % Esta rotina permite a criação do modelo estrutural
3 % por meio de uma entrada numérica.
4
5 % Número de nós e elementos
6 n_el=30;
7 n_nos=22;
8 n_pilares=20;
9 n_vigas = n_el - n_pilares;
10 pilaresporpav = 2;
11
12 % Criação de espaço para os vetores
13 node_info{n_nos,3} = [];
14 elem_info{n_el,:} = [];
15 sect_info = zeros(n_el,12);
16 sect_name{:,1} = [];
17 mat_info{:,4} = [];
18 mat_name{:,1} = [];
19 nload_info{n_nos,:} = [];
20 uniload_info{n_el,:} = [];
21 fixity_info{n_nos,:} = [];
22 settle_info{n_nos,:} = [];
23
24 %% Informações dos nós
25
26 % Coordenadas XYZ dos nós
27 % node_info = [coord_x coord_y coord_z]
28 node_info = [ 0    0    0;
29              30   0    0;
30              0  15    0;
31              30  15    0;
32              0  27    0;
33              30  27    0;
34              0  39    0;
35              30  39    0;
36              0  51    0;
37              30  51    0;
38              0  63    0;
39              30  63    0;
40              0  75    0;
41              30  75    0;
42              0  87    0;
43              30  87    0;
44              0  99    0;
45              30  99    0;
46              0 111    0;

```

```

47         30 111 0;
48         0 123 0;
49         30 123 0]*12;
50
51 % Condições de Suporte dos nós: Deslocamentos em XYZ e Rotação
    em XYZ
52 % Livre = NaN, Fixo = 0 ou valor
53 % support_info = [desloc_x desloc_y desloc_z rotação_x
    rotação_y rotação_z]
54 support_info = [0 0 NaN NaN NaN 0;
55                 0 0 NaN NaN NaN 0;
56                 NaN NaN NaN NaN NaN NaN;
57                 NaN NaN NaN NaN NaN NaN;
58                 NaN NaN NaN NaN NaN NaN;
59                 NaN NaN NaN NaN NaN NaN;
60                 NaN NaN NaN NaN NaN NaN;
61                 NaN NaN NaN NaN NaN NaN;
62                 NaN NaN NaN NaN NaN NaN;
63                 NaN NaN NaN NaN NaN NaN;
64                 NaN NaN NaN NaN NaN NaN;
65                 NaN NaN NaN NaN NaN NaN;
66                 NaN NaN NaN NaN NaN NaN;
67                 NaN NaN NaN NaN NaN NaN;
68                 NaN NaN NaN NaN NaN NaN;
69                 NaN NaN NaN NaN NaN NaN;
70                 NaN NaN NaN NaN NaN NaN;
71                 NaN NaN NaN NaN NaN NaN;
72                 NaN NaN NaN NaN NaN NaN;
73                 NaN NaN NaN NaN NaN NaN;
74                 NaN NaN NaN NaN NaN NaN;
75                 NaN NaN NaN NaN NaN NaN];
76
77
78 %% Informações dos elementos
79
80 % elem_info = [1-Nó i, 2-Nó j, 3-Seção #, ...
81 % 4-Material #, 5-Ângulo (rads), ...
82 % 6-Flexure condition Node i (rigid=0,pin=1,spring=2), ...
83 % 7-Flexure condition Node j (rigid=0,pin=1,spring=2), ...
84 % 8-Warping condition Node i (fixed=0,free=1,cont=2), ...
85 % 9-Warping condition Node j (fixed=0,free=1,cont=2), ...
86 % 10-Major-axis spring stiffness node i (val = 0 (pin) to inf (
    rigid)),...
87 % 11-Minor-axis spring stiffness node i (val = 0 (pin) to inf (
    rigid)),...
88 % 12-Major-axis spring stiffness node i (val = 0 (pin) to inf (
    rigid)),...
89 % 13-Minor-axis spring stiffness node i (val = 0 (pin) to inf (
    rigid)),...
90 % 14-Major-axis spring moment capacity Mpz node i (val = value
    to inf (unlimited)),...
91 % 15-Minor-axis spring moment capacity Mpz node i (val = value

```

```

    to inf (unlimited)),...
92 % 16-Major-axis spring moment capacity Mpz node j (val = value
    to inf (unlimited)),...
93 % 17-Minor-axis spring moment capacity Mpz node j (val = value
    to inf (unlimited))]
94 elem_info = [1      3      1      1 0 0 0 0 0 inf inf inf inf inf inf
    inf inf;
95 2      4      1      1 0 0 0 0 0 inf inf inf inf inf inf inf;
96 3      5      1      1 0 0 0 0 0 inf inf inf inf inf inf inf;
97 4      6      1      1 0 0 0 0 0 inf inf inf inf inf inf inf;
98 5      7      2      1 0 0 0 0 0 inf inf inf inf inf inf inf;
99 6      8      2      1 0 0 0 0 0 inf inf inf inf inf inf inf;
100 7     9      2      1 0 0 0 0 0 inf inf inf inf inf inf inf;
101 8     10     2      1 0 0 0 0 0 inf inf inf inf inf inf inf;
102 9     11     3      1 0 0 0 0 0 inf inf inf inf inf inf inf;
103 10    12     3      1 0 0 0 0 0 inf inf inf inf inf inf inf;
104 11    13     3      1 0 0 0 0 0 inf inf inf inf inf inf inf;
105 12    14     3      1 0 0 0 0 0 inf inf inf inf inf inf inf;
106 13    15     4      1 0 0 0 0 0 inf inf inf inf inf inf inf;
107 14    16     4      1 0 0 0 0 0 inf inf inf inf inf inf inf;
108 15    17     4      1 0 0 0 0 0 inf inf inf inf inf inf inf;
109 16    18     4      1 0 0 0 0 0 inf inf inf inf inf inf inf;
110 17    19     5      1 0 0 0 0 0 inf inf inf inf inf inf inf;
111 18    20     5      1 0 0 0 0 0 inf inf inf inf inf inf inf;
112 19    21     5      1 0 0 0 0 0 inf inf inf inf inf inf inf;
113 20    22     5      1 0 0 0 0 0 inf inf inf inf inf inf inf;
114 3     4      6      1 0 0 0 0 0 inf inf inf inf inf inf inf;
115 5     6      6      1 0 0 0 0 0 inf inf inf inf inf inf inf;
116 7     8      6      1 0 0 0 0 0 inf inf inf inf inf inf inf;
117 9     10     7      1 0 0 0 0 0 inf inf inf inf inf inf inf;
118 11    12     7      1 0 0 0 0 0 inf inf inf inf inf inf inf;
119 13    14     7      1 0 0 0 0 0 inf inf inf inf inf inf inf;
120 15    16     8      1 0 0 0 0 0 inf inf inf inf inf inf inf;
121 17    18     8      1 0 0 0 0 0 inf inf inf inf inf inf inf;
122 19    20     8      1 0 0 0 0 0 inf inf inf inf inf inf inf;
123 21    22     9      1 0 0 0 0 0 inf inf inf inf inf inf inf];
124
125 % Carregamentos uniformemente distribuídos nos elementos
126 % uniload_info = [carreg_x  carreg_y  carreg_z]
127 uniload_info = [0      0      0;
128                 0      0      0;
129                 0      0      0;
130                 0      0      0;
131                 0      0      0;
132                 0      0      0;
133                 0      0      0;
134                 0      0      0;
135                 0      0      0;
136                 0      0      0;
137                 0      0      0;
138                 0      0      0;
139                 0      0      0;

```

```

140         0     0     0;
141         0     0     0;
142         0     0     0;
143         0     0     0;
144         0     0     0;
145         0     0     0;
146         0     0     0;
147         0 -6/12  0;
148         0 -6/12  0;
149         0 -6/12  0;
150         0 -6/12  0;
151         0 -6/12  0;
152         0 -6/12  0;
153         0 -6/12  0;
154         0 -6/12  0;
155         0 -6/12  0;
156         0 -3/12  0];
157
158 % Efeitos da temperatura
159 thermal_info = zeros(n_el,4);
160
161 %% Informações da Seção
162
163 % sect_info = [1-Área, 2-Momento de inércia Izz, ...
164 % 3-Momento de inércia Iyy,...
165 % 4-Momento torsor J, 5-Coeficiente Cw, ...
166 % 6-Módulo de seção plástica Zzz, 7-Módulo de seção plástica Zyy
167 % 8-Área de cisalhamento Ayy, 9-Área de cisalhamento Azz, ...
168 % 10-YldSurf(1) 11-YldSurf(2) 12-YldSurf(3)]
169
170 global W;
171 global W12e14;
172
173 sect_info2 = zeros(n_el,3);
174 L = zeros(n_el,1);
175 i_perfil2 = zeros(n_el,1);
176
177 for el=1:n_el
178     i_perfil = perfis(elem_info(el,3));
179
180     if el<=n_pilares
181         sect_info(el,:) = [W12e14(i_perfil,1) W12e14(i_perfil,9)
182             W12e14(i_perfil,13) W12e14(i_perfil,17) W12e14(
183                 i_perfil,18) W12e14(i_perfil,10) W12e14(i_perfil
184                     ,14) Inf Inf 1 1 1];
185         sect_info2(el,:)=[W12e14(i_perfil,12) W12e14(i_perfil
186             ,16) W12e14(i_perfil,21)];
187         L(el,:)=sqrt((node_info(elem_info(el,2),1) - node_info(
188             elem_info(el,1),1))^2 + (node_info(elem_info(el,2)
189                 ,2) - node_info(elem_info(el,1),2))^2);
190         i_perfil2(el,:)=perfis(elem_info(el,3));

```

```

185     else
186         sect_info(el,:) = [W(i_perfil,1) W(i_perfil,9) W(
            i_perfil,13) W(i_perfil,17) W(i_perfil,18) W(
            i_perfil,10) W(i_perfil,14) Inf Inf 1 1 1];
187         sect_info2(el,:)=[W(i_perfil,12) W(i_perfil,16) W(
            i_perfil,21)];
188         L(el,:)=sqrt((node_info(elem_info(el,2),1) - node_info(
            elem_info(el,1),1))^2 + (node_info(elem_info(el,2)
            ,2) - node_info(elem_info(el,1),2))^2);
189         i_perfil2(el,:)=perfis(elem_info(el,3));
190     end
191 end
192
193 % Cargas nocionais
194 for el=1:n_el
195     cargas_gravitacionais(el,:)=[unload_info(el,2)*L(el,:)];
196     cargas_gravitacionais_peso(el,:)=[sect_info(el,1)*0.000284*L(el
            ,:)/12];
197     cargas_nocionais(el,:)=[0.002*(-cargas_gravitacionais(el,:)+
            cargas_gravitacionais_peso(el,:))];
198 end
199
200 % Carregamentos aplicados nos nós
201 % nload_info = [força_x força_y força_z momento_x momento_y
            momento_z]
202 nload_info = [0    0    0    0    0    0
203              0    0    0    0    0    0
204              10+cargas_nocionais(21,1)    0    0    0    0    0
205              0    0    0    0    0    0
206              10+cargas_nocionais(22,1)    0    0    0    0    0
207              0    0    0    0    0    0
208              10+cargas_nocionais(23,1)    0    0    0    0    0
209              0    0    0    0    0    0
210              10+cargas_nocionais(24,1)    0    0    0    0    0
211              0    0    0    0    0    0
212              10+cargas_nocionais(25,1)    0    0    0    0    0
213              0    0    0    0    0    0
214              10+cargas_nocionais(26,1)    0    0    0    0    0
215              0    0    0    0    0    0
216              10+cargas_nocionais(27,1)    0    0    0    0    0
217              0    0    0    0    0    0
218              10+cargas_nocionais(28,1)    0    0    0    0    0
219              0    0    0    0    0    0
220              10+cargas_nocionais(29,1)    0    0    0    0    0
221              0    0    0    0    0    0
222              5+cargas_nocionais(30,1)    0    0    0    0    0
223              0    0    0    0    0    0];
224
225 secoes = zeros(n_el,9);
226 for el=1:n_el
227     siz=size(ele_for);
228     %secoes=el i_perfil A E Ix rx ry L peso linear

```

```

229     secoes(el,:)=[el i_perfil2(el,1) sect_info(el,1) mat_info(
        elem_info(el,4) sect_info(el,2) sect_info2(el,1)
        sect_info2(el,2) L(el,1) sect_info2(el,3)];
230 end
231
232 % Nomes das seções
233 sect_name = {'Seção 1';...
234             'Seção 2';...
235             'Seção 3';...
236             'Seção 4';...
237             'Seção 5';...
238             'Seção 6';...
239             'Seção 7';...
240             'Seção 8';...
241             'Seção 9'};
242
243 %% Informações do material
244
245 % mat_info = [Módulo de elasticidade E, Poisson v, ...
246 % Tensão de escoamento Fy, Peso específico]
247 % Redução da rigidez para consideração da não-linearidade
    física
248 mat_info = [0.8*29000 0.3 36 0.00284];
249
250 % Nome do material
251 mat_name = {'A36'};
252
253 %% Fim da entrada de dados
254
255 %% Início da rotina de cálculo
256
257 webdir = el_webdir(size(elem_info,1),elem_info(:,1:2),elem_info
    (:,5),node_info,node_info,zeros(size(node_info,1),6));
258
259 elem_info = [ elem_info(:,1:5) ones(size(elem_info,1),2)
    webdir ones(size(elem_info,1),1) ...
260             elem_info(:,6:7) ones(size(elem_info,1),2) ...
261             elem_info(:,8:9) ones(size(elem_info,1),2) ...
262             elem_info(:,10:17)];
263 nload_info = [nload_info NaN(size(nload_info,1),6)];
264 fixity_info = support_info;
265 fixity_info(find(~isnan(support_info))) = 0;
266 fixity_info = [fixity_info NaN(size(fixity_info,1),6)];
267 settle_info = support_info;
268 settle_info(find(~isnan(settle_info)& settle_info==0)) = NaN;
269 settle_info = [settle_info NaN(size(settle_info,1),6)];
270 uniload_info = [uniload_info zeros(size(uniload_info,1),3) NaN(
    size(uniload_info,1),6) thermal_info];
271
272 %% Configurações
273
274 h_stat_mes = [];

```



```

275 struct_type = 2; %Pórtico espacial(1)|Pórtico plano(2)
276 E_t         = 1; %E(1)
277 anatype     = 2; %Primeira ordem elástica(1)|Segunda ordem
           elástica(2)
278 sol_scheme  = 2; %Simple Step(1)|Predictor-Corrector(2)|
279 incr_size   = 0.1; %Tamanho do incremento
280 incr_nums   = 10; %Número de incrementos
281 stop_ratio  = 1; %Soma dos incrementos totais
282 restart     = 1; %Começar uma nova análise(1)
283 apratios    = [];
284 user_def    = 0; %Códigos do Mastan2(0)
285
286 % Chama a rotina de análise
287 analysis_info=[];periods_info=[];stiff=[];ele_pldef=[];
288 defl=[];react=[];ele_for=[];ele_yld=[];applfm=[];
289
290 [analysis_info,periods_info,stiff,ele_pldef,defl,react,ele_for,
           ele_yld,...
291  applfm] = ud_batch_anaprep(h_stat_mes,struct_type,E_t,...
292  anatype,sol_scheme,incr_size,incr_nums,stop_ratio,restart,
           apratios,...
293  user_def,node_info,elem_info,sect_info,mat_info,fixity_info
           ,...
294  nload_info,settle_info,unload_info,analysis_info,
           periods_info,...
295  stiff,ele_pldef,defl,react,ele_for,ele_yld,applfm);
296 apratios = analysis_info(5:end)';
297
298 % Restrição de deslocamento
299 % Pega os valores de desloc. na extremidade direita e subtrai 2
           a 2 cada nó
300 for nos=1:n_nos
301     desloc(nos,:)= [defl(nos,1,siz(1,3)) defl(nos,2,siz(1,3))
           defl(nos,6,siz(1,3))];
302 end
303 auxiliar = desloc';
304 interstorydrift = diff(auxiliar(1,pilaresporpav:pilaresporpav
           :end));
305 andares = diff(node_info((pilaresporpav:pilaresporpav:end),2));
306 h_andares = andares';
307 restricao_desloc = sum((interstorydrift ./ (h_andares/300))-1);
308
309 % Vetor dos coeficientes de comprimento efetivo
310 el_conectados= containers.Map('KeyType','double', 'ValueType','
           any');
311
312 for el=1:n_el
313     %será usado para o cálculo de G (função acharG)
314     if isKey(el_conectados,conec(el,3))
315         el_conectados(conec(el,3)) = [el_conectados(conec(el,3))
           el];
316     else

```

```

317     el_conectados(conec(el,3)) = el;
318     end
319     if isKey(el_conectados,conec(el,4))
320         el_conectados(conec(el,4)) = [el_conectados(conec(el,4))
321             el];
322     else
323         el_conectados(conec(el,4)) = el;
324     end
325 end
326 function G = acharG(vetor)
327     if numel(vetor)==1
328         G= 10;
329     else
330         ehpillar = vetor(vetor <= n_pilares);
331         ehviga = vetor(vetor > n_pilares);
332         Ip = secoes(ehpillar,5);
333         Lp = secoes(ehpillar,8);
334         Iv = secoes(ehviga,5);
335         Lv = secoes(ehviga,8);
336         G = sum(Ip./Lp)/sum(Iv./Lv);
337     end
338 end
339
340 vetorG = zeros(1,n_nos);
341 for no=1:n_nos
342     conectados = el_conectados(no);
343     vetorG(no) = acharG(conectados);
344 end
345
346 function kx = acharkx (Ga,Gb)
347     kx = sqrt((1.6*Ga*Gb+4*(Ga+Gb)+7.5)/(Ga+Gb+7.5));
348 end
349 vetorKx = zeros(n_el,1);
350
351 Cbs = zeros(1,n_el);
352 for i=1:n_el
353     if i<=n_pilares %pilares - momento linear
354         Mi = ele_for(i,6,siz(1,3));
355         Mf = ele_for(i,12,siz(1,3));
356         Mb = (Mi+Mf)/2;
357         Ma = (Mi+Mb)/2;
358         Mc = (Mb+Mf)/2;
359         Mmax = max(abs(Mi),abs(Mf));
360         Cbs(i) = 12.5*Mmax/(2.5*Mmax+3*abs(Ma)+4*abs(Mb)+3*abs(
361             Mc));
362     else %vigas - momento quadrático
363         Mi = ele_for(i,6,siz(1,3));
364         Mf = abs(ele_for(i,12,siz(1,3)));
365         carga_q = -(unload.info(i,2));
366         %momento no vão calculado pelo cortante M1 - V^2/(2*q)
367         M_vao(i,:) = abs(Mi - ele_for(i,2,siz(1,3))^2/carga_q/2)

```

```

;
367     Ma = abs(Mi-ele_for(i,2,siz(1,3))*secoes(el,8)*0.25 +
        carga_q/2*(secoes(el,8)*0.25)^2);
368     Mb = abs(Mi-ele_for(i,2,siz(1,3))*secoes(el,8)*0.5 +
        carga_q/2*(secoes(el,8)*0.5)^2);
369     Mc = abs(Mi-ele_for(i,2,siz(1,3))*secoes(el,8)*0.75 +
        carga_q/2*(secoes(el,8)*0.75)^2);
370     Mmax = max([Mi,Mf,M_vao(i)]);
371     Cbs(i) = 12.5*Mmax/(2.5*Mmax+3*abs(Ma)+4*abs(Mb)+3*abs(
        Mc));
372     end
373 end
374     Pu = zeros(n_el,1);
375     Mu = zeros(n_el,1);
376     for el=1:n_el
377         Pu(el,:)=ele_for(el,7,siz(1,3));
378         Mu(el,:) = max(abs(ele_for(el,[6,12],siz(1,3)))));
379         Mu(el,:) = max(Mu(el,:),M_vao(el,:));
380     end
381 end

```

A.2 ANÁLISE ESTRUTURAL DO PÓRTICO

```

1 function [analysis_info,periods_info,stiff,ele_pldef,defl,react,
        ele_for,ele_yld,...
2     applfm,status_mes] = ud_batch_anaprep(h_stat_mes,struct_type
        ,E_t,...
3     anatype,sol_scheme,incr_size,incr_nums,stop_ratio,restart,
        apratios,...
4     user_def,node_info,elem_info,sect_info,mat_info,fixity_info
        ,...
5     nload_info,settle_info,uniload_info,analysis_info,
        periods_info,...
6     stiff,ele_pldef,defl,react,ele_for,ele_yld,applfm)
7
8 % Função do MASTAN2 modificada para a utilização com o SGA
9 % Autores: R.D. Ziemian (Bucknell University)
10 % e W. McGuire (Cornell University)
11
12 % Dados necessários para a rotina de análise
13 node_total = size(node_info,1);
14 elem_total = size(elem_info,1);
15
16 truss = 0;
17 flag = 0;
18 nele = elem_total;
19 nnodes = node_total;
20 coord = node_info(1:nnodes,1:3);
21 ends = elem_info(1:nele,[1 2 12 13 16 17 20 21 22 23 24 25 26

```

```

22     27]);
23
24     for i = 1:nnodes
25         for j = 1:6
26             if isnan(fixity_info(i,j)) & isnan(settle_info(i,j));
27                 fixity(i,j) = NaN;
28             elseif ~isnan(fixity_info(i,j)) & isnan(settle_info(i,j)
29                 )
30                 fixity(i,j) = fixity_info(i,j);
31             else
32                 fixity(i,j) = settle_info(i,j);
33             end
34         end
35     end
36     A = sect_info(elem_info(1:nele,3),1);
37     Zzz = sect_info(elem_info(1:nele,3),6);
38     Zyy = sect_info(elem_info(1:nele,3),7);
39     if ~truss
40         for ele = 1:nele
41             if elem_info(1:nele,12)== 1 & elem_info(1:nele,13)== 1
42                 Izz(ele) = 0;
43                 Iyy(ele) = 0;
44                 Ayy(ele,1) = inf;
45                 Azz(ele,1) = inf;
46             else
47                 Izz(ele) = sect_info(elem_info(ele,3),2);
48                 Iyy(ele) = sect_info(elem_info(ele,3),3);
49                 Ayy(ele) = sect_info(elem_info(ele,3),8);
50                 Azz(ele) = sect_info(elem_info(ele,3),9);
51             end
52         end
53         J = sect_info(elem_info(1:nele,3),4);
54         Cw = sect_info(elem_info(1:nele,3),5);
55     end
56
57     YldSurf = sect_info(elem_info(1:nele,3),10:12);
58     E = mat_info(elem_info(1:nele,4),1);
59     v = mat_info(elem_info(1:nele,4),2);
60     Fy = mat_info(elem_info(1:nele,4),3);
61     Wt = mat_info(elem_info(1:nele,4),4);
62     webdir = elem_info(1:nele,8:10);
63     concen = nload_info(1:nnodes,1:6);
64     w = uniload_info(1:nele,1:3);
65     thermal = uniload_info(1:nele,13:16);
66
67     for ele = 1:nele
68         del(ele,:) = coord(ends(ele,2),:)-coord(ends(ele,1),:);
69         L(ele) = norm(del(ele,:),2);
70     end
71

```

```

72 if struct_type == 2
73     two_dim = 1;
74 end
75
76 if two_dim == 1
77     fixity(1:nnodes,3:5) = 0;
78     ends(1:nele,5:6) = 1;
79 end
80
81 if anatype == 2
82     if size(analysis_info,2) > 4
83         if two_dim ~= analysis_info(1) | truss ~= analysis_info
84             (2) | anatype ~= analysis_info(3)
85             apratios = [];
86             limit_state = 0;
87         end
88         if size(apratios,1) > 0
89             limit_state = analysis_info(4);
90         else
91             limit_state = 0;
92         end
93     else
94         limit_state = 0;
95     end
96
97     if restart == 1 | isempty(apratios)
98         total_pre = 0;
99         applfm = [];
100     else
101         total_pre = size(apratios,1);
102     end
103
104 % Chama código que contém a rotina de análise
105 if user_def == 0
106     [stiff,defl,react,ele_for,aflag,apratios,limit_state] =
107         msa_2el(nnodes,coord,concen,fixity,nele,ends,...
108             A,Izz,Iyy,J,Cw,Zzz,Zyy,Ayy,Azz,E,v,Fy,YldSurf,Wt
109             ,webdir,beta_ang,w,thermal,truss,E,t,
110             anatype,sol_scheme,incr_nums,incr_size,...
111             stop_ratio,restart,stiff,defl,react,ele_for,
112             apratios,limit_state,h_stat_mes);
113
114     ele_yld = zeros(nele,2,size(ele_for,3));
115     ele_pldef = zeros(nele,14,size(ele_for,3));
116
117     total = size(apratios,1);
118     if total == 0
119         load_val = 0;
120     else
121         load_val = apratios(total);
122     end
123 end

```

```

119     if total > 0
120         for iii=total_pre+1:total
121             if iii > 1
122                 applfm(:, :, iii) = applfm(:, :, iii-1) + (apratios(
                    iii)-apratios(iii-1))*concen;
123             else
124                 applfm(:, :, 1) = apratios(iii)*concen;
125             end
126         end
127     else
128         applfm = [];
129     end
130
131     % Apaga figuras criadas pelo código do Mastan2
132     delete(h.fig);
133 end

```

A.3 CÁLCULO DAS RESISTÊNCIAS

```

1 function [Pn,Mn,resist_tracao] = ...
2     esforcos.nominais(secoes,Cbs,vetorKx,n_pilares,mat_info)
3 E=mat_info(1,1);
4 Fy=mat_info(1,3);
5 refy = sqrt(E/Fy);
6
7 global W;
8 global W12e14;
9
10 n_el = numel(Cbs);
11 n_vigas = n_el - n_pilares;
12
13 % Itera todos os pilares e vigas e
14 % computa a resistência dos elementos.
15 % Usa uma função que recebe o índice e calcula a resistência.
16 % Retorna:
17 % vetor Pn_viga e Mn_viga
18 % vetor Pn_pilar e Mn_pilar
19 % Pn = [Pn_pilar Pn_viga];
20 % Mn = [Mn_pilar Mn_viga];
21
22 %% Cálculo da resistência à tração
23 resist_tracao = secoes(:,3)*Fy;
24
25 %% Recebe a seção do pilar e da viga e calcula o kx de cada
    elemento.
26
27 ky_pilares = 1;
28 ky_vigas   = 0.2;
29 %Constroi vetorKy para ky constante entre vigas e pilares

```

```

30 vetorKy = [ky_pilares*ones(n_pilares,1);ky_vigas*ones(n_vigas,1)
   ];
31
32 %Verifica a máxima relação KL/r entre eixo X e Y
33 klr = max(vetorKx./secoes(:,6), vetorKx./secoes(:,7)).*secoes
   (:,8);
34
35 function [Pn,Mn] = achar_resistencia(indice, klr, L, ky, Cb,el)
36 %% Propriedades da seção
37 if el<=n_pilares
38     A = W12e14(indice,1);
39     d = W12e14(indice,2);
40     tw = W12e14(indice,3);
41     bf = W12e14(indice,4);
42     tf = W12e14(indice,5);
43     bt = W12e14(indice,7);
44     htw = W12e14(indice,8);
45     Ix = W12e14(indice,9);
46     Zx = W12e14(indice,10);
47     Sx = W12e14(indice,11);
48     rx = W12e14(indice,12);
49     Iy = W12e14(indice,13);
50     Zy = W12e14(indice,14);
51     Sy = W12e14(indice,15);
52     ry = W12e14(indice,16);
53     J = W12e14(indice,17);
54     Cw = W12e14(indice,18);
55     h0 = W12e14(indice,19);
56     rts = W12e14(indice,20);
57
58 else
59     A = W(indice,1);
60     d = W(indice,2);
61     tw = W(indice,3);
62     bf = W(indice,4);
63     tf = W(indice,5);
64     bt = W(indice,7);
65     htw = W(indice,8);
66     Ix = W(indice,9);
67     Zx = W(indice,10);
68     Sx = W(indice,11);
69     rx = W(indice,12);
70     Iy = W(indice,13);
71     Zy = W(indice,14);
72     Sy = W(indice,15);
73     ry = W(indice,16);
74     J = W(indice,17);
75     Cw = W(indice,18);
76     h0 = W(indice,19);
77     rts = W(indice,20);
78 end
79

```

```

80
81 %% Compressão - sem elementos esbeltos
82 % Cálculo padrão usado para o caso de elementos esbeltos
83 Fe = pi^2*E/klr^2;
84 if klr <= 4.71*refy
85     Fcr = 0.658^(Fy/Fe)*Fy;
86 else
87     Fcr = 0.877*Fe;
88 end
89
90 %% Compressão - com elementos esbeltos
91 % Verificação da esbeltez na compressão da alma
92 % Nota: todas as mesas da série W são compactas para
      compressão
93 Qa=1;
94 if htw > 1.49*refy % h/tw > lim alma não compacta
95     ref = sqrt(E/Fcr);
96     if htw >= 1.49*ref
97         be = 1.92 * tw * ref*(1- 0.34/htw*ref); % nova alma
          -> H novo
98         if be < htw*tw
99             Aeff = A - tw*(htw*tw - be); % novoA = velhoA -
          tw*redução
100            Qa = Aeff/A;
101        end
102    end
103 end
104
105 %% Cálculo de Pn
106 % Alteração de Fcr com base na esbeltez
107 if Qa==1
108     Pn = Fcr*A;
109 else
110     if Fe >= 0.44*Qa*Fy
111         Fcr = Qa*(0.658^(Qa*Fy/Fe))*Fy;
112     else
113         Fcr = 0.877*Fe;
114     end
115     Pn = Fcr*A;
116 end
117
118 %% Flexão
119 % Nota: somente W6x15 tem mesa não-compacta na série W
120 % Considerando fy=36ksi
121 % Todos os outros tem mesa compacta na flexão
122 % Além disso, todas as almas são compactas na flexão
123 % Resultando em dois casos:
124 % 1) alma e mesa compactas -- F2.[1,2]
125 % 2) alma compacta e mesa não compacta (W6x15) -- F2.2 e F3
      .2
126
127 % Caso 1 - alma e masa compactas

```



```

128     % a) Plastificação da seção transversal
129     Mp = Zx*Fy;
130
131     % b) Flambagem lateral com torção (FLT)
132     Lb = ky*L; % Considera o contraventamento
133
134     Lp = 1.76*ry*refy; %resultado em polegadas
135     Lr = 1.95*rts*E/(0.7*Fy)*sqrt(J/(Sx*h0) + sqrt(J/(Sx*h0)
136         ^2...
137         + 6.76*(0.7*Fy/E)^2));
138
139     if Lb <= Lp
140         Mn = Mp;
141     elseif (Lp < Lb) && (Lb <= Lr)
142         Mn = Cb*(Mp - (Mp - 0.7*Fy*Sx)*(Lb-Lp)/(Lr-Lp));
143     else
144         Fcr = Cb*pi^2*E/(Lb/rts)^2*sqrt(1+0.078*J/(Sx*h0)*(Lb/
145             rts)^2);
146         Mn = Fcr*Sx;
147     end
148
149     % Caso 2 - alma compacta e mesa não compacta (W6x15)
150     if indice == 6
151         lamb_p = 0.38*refy;
152         lamb_r = refy;
153         Mn = Mp - (Mp-0.7*Fy*Sx)*(bt-lamb_p)/(lamb_r-lamb_p);
154     end
155
156     % Mn nunca pode ser maior que Mp (escoamento da seção bruta)
157     if Mn > Mp
158         Mn = Mp;
159     end
160 end
161
162 %Pré-alocação do tamanho do vetor
163 Pn = zeros(1,n-el);
164 Mn = zeros(1,n-el);
165
166 for el=1:n-el
167     [Pn(el),Mn(el)] = achar_resistencia(secoes(el,2),klr(el),...
168         secoes(el,8),vetorKy(el),Cbs(el),el);
169 end
170 end

```

A.4 FUNÇÃO OBJETIVO

```

1 function [ obj ] = fobj_portico(perfis)
2 % Função objetivo recebe índices da viga e pilar:
3 % Retorna peso se ok ou então peso + penalidade

```

```

4 perfis = round(perfis);
5
6 %% Esforços requeridos
7 % Chama código de pórtico plano e retorna com a matriz de forças
  e momentos
8 % em cada elemento do pórtico
9
10 [Pu,Mu,Cbs,secoes,vetorKx,n.pilares,restricao_desloc] =
  Portico.nt_lt(perfis);
11
12
13 %% Esforços nominais - resistência
14 % Computa a resistência do perfil
15 % Depende de KL/r, logo varia para cada elemento
16
17 [Pr,Mr,resist_tracao] = esforcos.nominais(secoes,Cbs,vetorKx,
  n.pilares,mat.info);
18 Pr = Pr';
19 Mr = Mr';
20
21 % A função portico.nt_lt calcula todos os elementos como
  flexocompressão,
22 % caso o esforço em algum elemento seja de tração, este "for"
  coloca a
23 % resistência do perfil correspondente
24 for i=find(Pu>0)
25     Pr(i) = resist_tracao(i);
26 end
27
28 % Feita a consideração dos sinais (Tr/Comp), pode-se pegar o
  valor em
29 % módulo apenas
30 Pu = abs(Pu);
31
32 n_el=numel(Pu);
33 interacao = zeros(1,n_el);
34 for i=1:n_el
35     UsobreR = Pu(i)/(0.9*Pr(i));
36     if UsobreR >= 0.2
37         interacao(i) = fix((UsobreR + 8/9*(Mu(i)/(0.9*Mr(i))))
          *100)/100;
38     else
39         interacao(i) = fix((UsobreR/2 + Mu(i)/(0.9*Mr(i)))*100)
          /100;
40     end
41 end
42
43 % Peso = soma de peso linear * L
44 peso = sum(secoes(:,9).*secoes(:,8)/12);
45
46 % Resultado da interação de flexo-compressão, deve ser menor que
  1

```

```
47 % 0 para ok ou o quanto excedeu de 1.0
48 restricao1 = interacao-1;
49 restricao2 = restricao_desloc;
50 P = sum(max(0,[restricao1 restricao2]));
51 alfa = 10^10;
52
53 obj = peso + alfa * P;
54 end
```


APÊNDICE B – Rotina computacional - *SGA*

B.1 ROTINA PRINCIPAL

```

1 function [ minimo,coordenadas,dadositeration,diversidade] = RGA(
    F,alfa0,alfamin)
2 % Programado para encontrar o conjunto de coordenadas, contido
    dentro dos
3 % limites de lsup e linf, ao qual minimizam o valor de fobj,
    quando
4 % comparados com qualquer outro conjunto de coordendas que
    respeite as
5 % mesmas restrições
6 % lsup = limite superior do domínio analisado
7 % linf = limite inferior do domínio analisado
8 % alfa0 = porcentagem de aleatoriedade principal
9 % alfamin = porcentagem minima de aleatoriedade, visa garantir
    que a
10 % aleatoriedade nunca zere
11 % funobj = função a ser otimizada
12
13 format long
14
15 %% Parametros do Otimizador
16 % Aqui são inicializados todos os parâmetros necessários para o
17 % funcionamento do algoritmo, cada qual sera explicado em
    particular;
18
19 [linf,lsup,fobj,dim] = fobj(F);
20
21 % Define a porcentagem de individuos que, pelo seu rank,
    garantem vaga
22 % no grupo de otimização sem necessitar participar do torneio
23 elite = 1;
24
25 % População
26 n = 50;
27
28 % Número máximo de iterações
29 nmax = 150;
30
31 % Define o tamanho do grupo de otimização, representa a
    porcentagem do
32 % tamanho destecom relação a população total
33 ng=0.2;
34
35 % Porcentagem das iterações dedicadas à otimização global
36 nglobal = 0.5;
37
38 % Porcentagem das iterações dedicadas à otimização local
39 % nlocal = 1 - nglobal;

```

```

40 nlocal = round(100*(1-nglobal))/100;
41
42 % Quantidade de individuos inseridos no processo de otimização,
    por meio de
43 % perturbação relacionada a media e desvio padrão do grupo total
44 npertub = 5;
45
46 % Porcentagem do alfaminimo que atuara como limitante inferior
    na
47 % aleatoriedade na etapa de otimização local
48 residuominimo = 0.007;
49
50 % Coeficientes linear e angular, respectivamente, das retas que
    definem o
51 % decaimento da aleatoriedade com o número de iterações,
    referentes à etapa
52 % de otimização global
53 matrizaleatoriedade = [1 -4/(nmax*nglobal);0.25 -1/(4*nmax*
    nglobal);0 0];
54
55 % Número de retas utilizadas para definir o decaimento da
    aleatoriedade
56 % descrito anteriormente
57 numeroderetas=size(matrizaleatoriedade);
58
59 % Define o número de indivíduos que se enfrentara em cada etapa
    do torneio
60 tamanhotorneio = 4;
61
62 % Define a aleatoriedade de cada iteração, representa a
    amplitude do domínio
63 % ao qual cada indivíduo pertencente ao grupo de otimização pode
    gerar um descendente
64 alfa = (alfa0+alfamin)*(lsup-linf);
65
66 if length(lsup) ~= length(linf)
67     disp('Dimensões inválidas');
68 else %verifica se os limitantes do domínio tem as mesmas
    dimensões
69 %% Geração da população inicial
70
71 % Após a inicialização do algoritmo, este começa seu processo de
72 % otimização gerando a população inicial, de maneira aleatória
    em
73 % qualquer posição do domínio
74
75 x = bsxfun(@plus,linf,bsxfun(@times,lsup-linf,rand(n,dim)));
76 x = round(x);
77
78 %% Avalia fitness da população
79
80 % Avalia o valor da função objetivo em cada indivíduo da

```



```

    população
81 % inicial. A matriz bancodedados armazena todos os dados
    referentes as
82 % coordenadas e avaliação da função objetivo de todos os
    indivíduos
83 fertilidade = zeros(n,1);
84 for i = 1:n
85     fertilidade(i,1) = fobj(x(i,:));
86 end
87
88 bancodedados = [fertilidade x];
89 bancodedados = sortrows(bancodedados,1);
90
91
92 %% Seleção do primeiro grupo de otimização
93
94 % Aqui inicia-se o processo de seleção de indivíduos para
    participarem
95 % do grupo de otimização, que tem por responsabilidade gerar os
96 % indivíduos para as próximas etapas do processo de otimização.
    O vetor
97 % índices contem o índice de cada individuo selecionado. Uma
    parte esta
98 % selecionada diretamente pelo rank, definidos pelo parâmetro
    elite.
99 % Outra parcela é selecionada pelo algoritmo de torneio
100
101     indices = (1:n*ng)';
102     indicestorneio = torneio(bancodedados(:,1),n*(1-elite)*ng,
        tamanhotorneio);
103     dadositeration= zeros(nmax,dim+1);
104     indicestorneio = sort(indicestorneio);
105     indices(elite*ng*n+1:n*ng) = indicestorneio(:,1);
106
107 %% Formação do primeiro grupo de otimização
108
109 % Aqui efetivamente se monta o grupo de otimização, selecionando
    os
110 % membros pelos índices determinados anteriormente
111 cresceram = zeros(n*ng,dim+1);
112 for i = 1:n*ng
113     local = indices(i);
114     cresceram(i,:) = bancodedados(local,:);
115 end
116
117
118 %% Início do processo iterativo
119
120 %Inicia-se a seguir o processo iterativo de otimização.
    Primeiramente
121 %inicia-se pelo otimização global, onde busca-se explorar o
    máximo possível o domínio.

```

```

122 %Após isso inicia-se uma etapa local, onde busca-se otimizar
    ainda
123 %mais a função objetivo nas proximidades do ponto ótimo até
    então
124
125 %% Processo de otimização global
126
127 %No processo de otimização global, cada indivíduo do grupo de
128 %otimização pode gerar um determinado número de outros
    indivíduos. A
129 %quantidade de filhos que ele pode ter é função do seu rank no
    grupo.
130 %Cada família acaba por ser avaliada e apenas o melhor indivíduo
    fara
131 %parte do próximo grupo de otimização, até terminarem as
    iterações.
132 %Após a geração de cada novo grupo de otimização, um determinado
    número
133 %de outros indivíduos substituem alguns membros ja pertencentes
    deste
134 %grupo. O número de indivíduos que farão isto é definido pelo
    parâmetro
135 %npertub, sendo estes inseridos em função da média e desvio
    padrão do
136 %grupo inteiro.
137     diversidade = zeros(n,1);
138
139     disp('Fase Global')
140     for k = 1:ceil(nglobal*nmax)
141         indicespertub = torneioinverso(cresceram,npertub,
            tamanhotorneio);
142         %Seleciona os membros que serão substituidos, através de
            um torneio
143         %inverso, onde busca-se o perdedor para ser substituido
144         for t = 1:npertub
145             perturb = round(mean(cresceram(:,2:dim+1))) + t*
                round(std(cresceram(:,2:dim+1)).*(rand(1,dim)
                    -0.5));
146             perturb = max(perturb,linf);
147             perturb = min(perturb,lsup);
148             cresceram(indicespertub(t,1),2:dim+1) = perturb;
149         end
150
151 %% Processo de geração do grupo de otimização
152
153
154     [cresceram,diversity_index] = filhotes(cresceram,n,ng,
        lsup,linf,alfa,fobj);
155     cresceram = sortrows(cresceram,1);
156     dadositeration(k,:) = cresceram(1,:);
157     diversidade(k) = diversity_index;
158

```

```

159 %Aqui se utiliza o grupo de otimização da iteração anterior, ja
160 %saindo com o próximo grupo e a avaliação da função objetivo em
161 %cada um desses indivíduos. vale ressaltar que o ponto ótimo
    obtido
162 %até o momento nunca se perde, pois a participação dele no
    próximo
163 %grupo de otimização depende apenas da sua avaliação da função
164 %objetivo e este nunca sera substituido pelo torneio inverso,
    pois
165 %possui rank 1
166
167 %O dado do ponto ótimo até o momento é transferido para a matriz
168 %dadositeration, que contem todos os dados de cada iteração
169 %% Variação da aleatoriedade
170
171 %Aqui varia-se a aleatoriedade. O decaimento é definido pelas
    retas
172 %inicializadas anteriormente. Obtem-se um valor de ordenada
    entre 0
173 %e 1 para a matriz matrizusada, de acordo com o máximo da
    ordenada
174 %indicada por cada reta, em função da abscissa número de
    iterações.
175 %Este valor representa a porcentagem a porcentagem da
    aleatoriedade inicial que sera utilizada na próxima
    iteração.
176 %Soma-se a isso o alfamin, a fim de se garantir que nunca se
    tenha
177 %aleatoriedade zero, o que paralisaria o algoritmo.
178     matrizusada = zeros(numeroderetas(1,1),1);
179     for l = 1:numeroderetas(1,1)
180         matrizusada(l,1) = max(matrizaleatoriedade(l,1)+
            matrizaleatoriedade(l,2)*k);
181     end
182     alfa = (alfa0*max(matrizusada)+alfamin)*(lsup-linf);
183     disp(dadositeration(k,1));
184     disp(alfa(1,1));
185 end
186 %% Processo de otimização local
187
188 %Assemelha-se ao processo global, a principal diferença é que
    neste
189 %processo não diferenciam-se os individuos por familia, para
    formar o
190 %próximo grupo de otimização. Neste caso avaliam-se todos os
    indivíduos
191 %igualmente, passando para o próximo grupo os indivíduos melhor
192 %rankiados, independente da familia ao qual pertencem
193     disp('Fase Local')
194     for k = 1:ceil(nmax*nlocal)
195         indicespertub = torneioinverso(cresceram,npertub,
            tamanhotorneio);

```

```

196     for t = 1:npertub
197         perturb = round(mean(cresceram(:,2:dim+1))) + t*
                round(std(cresceram(:,2:dim+1)).*(rand(1,dim)
                -0.5));
198         perturb = max(perturb,linf);
199         perturb = min(perturb,lsup);
200         cresceram(indicespertub(t,1),2:dim+1) = perturb;
201         cresceram(indicespertub(t,1),1) = fobj(cresceram(
                indicespertub(t,1),2:dim+1));
202     end
203 %Processo de inserção de indivíduos idêntico ao processo que
    consta na etapa global
204
205     [bancodedados,diversity_index] = filhoteslocais(
        cresceram,n,ng,lsup,linf,alfa,fobj);
206     diversidade(round(nglobal*nmax)+k) = diversity_index;
207 %Esta função entra com o grupo de otimização desta iteração e
208 %retorna todos os indivíduos gerados e geradores, além de suas
209 %avaliações da função objetivo
210 %% Variação da aleatoriedade
211
212 %Varia de maneira semelhante à variação da etapa anterior.
    Difere
213 %por se tratar de apenas uma reta de decaimento para a
214 %aleatoriedade
215     alfa = (((nlocal*nmax-k)/(nlocal*nmax))*alfamin+
        residuominimo*alfamin)*(lsup-linf);
216 %Resíduo mínimo garante que a aleatoriedade não zere e também
217 %possibilita trabalhar com mínimos de aleatoriedade diferentes
    para
218 %a etapa global e local, bastando introduzir este parâmetro com
219 %valor diferente de 1
220     bancodedados = sortrows(bancodedados,1);
221 %Contém todos os dados de avaliação da função objetivo e
222 %coordenadas de cada indivíduo desta iteração
223     dadositeration(round(nglobal*nmax)+k,:) = bancodedados
        (1,:);
224 %O melhor indivíduo da iteração é anexado à matriz
    dadositeration
225
226     cresceram = zeros(ng*n,dim+1);
227     indices = (1:n*ng)';
228     indicestorneio = torneio(bancodedados(:,1),n*(1-elite)*
        ng,tamanhotorneio);
229     indicestorneio = sort(indicestorneio);
230     indices(elite*ng*n+1:n*ng) = indicestorneio(:,1);
231     for i = 1:n*ng
232         local = indices(i);
233         cresceram(i,:) = bancodedados(local,:);
234     end
235 %Monta-se o novo grupo de otimização, através dos membros
236 %elitizados (garantidos pelo seu rank) e daqueles vencedores do

```

```

237 %torneio
238     disp(alfa(1,1));
239     disp(dadositeration(round(100*(nglobal*nmax))/100+k));
240
241     end
242     coordenadas = zeros(1,dim);
243     coordenadas(1,:) = dadositeration(nmax,2:dim+1);
244 %Coordenadas do ponto ótimo obtido ao fim do processo de
    otimização
245     minimo = fobj(coordenadas);
246 %Ponto ótimo obtido ao fim do processo de otimização
247     dlmwrite('myFile2.txt',dadositeration,'-append','delimiter',
    ' ','roffset',1,'precision',8)
248     dlmwrite('2diversidade.txt',diversidade,'-append','delimiter
    ',' ','roffset',1,'precision',8)
249 end
250
251 end

```

B.2 CHAMADA DA FUNÇÃO OBJETIVO

```

1 function [lb,ub,fobj,dim] = fobj(F)
2
3 % lb é o limite inferior: lb=[lb_1,lb_2,...,lb_d]
4 % up é o limite superior: ub=[ub_1,ub_2,...,ub_d]
5 % dim é o número de variáveis
6 switch F
7     case 'portico'
8         fobj = @fobj_portico;
9         lb = [1 1 1 1 1 1 1 1 1];
10        ub = [66 66 66 66 66 66 267 267 267];
11        dim = 9;
12 end
13 end

```

B.3 CRIAÇÃO DAS FAMÍLIAS - ETAPA GLOBAL

```

1 function [ cresceram ] = filhotes( cresceram,n,ng,lsup,linf,alfa
    ,fobj)
2 % Função responsável por receber um dado grupo de otimização e
    retornar uma
3 % matriz contendo outro grupo de otimização, além da avaliação
    da função
4 % objetivo em cada indivíduo.
5
6 d = length(lsup);

```

```

7  indices = fazergrupos(n,ng);
8
9  for i = 1:ng*n
10     tamanho = indices(i,1);
11     ajuda = zeros(tamanho+1,d+1);
12     x = zeros(tamanho+1,d);
13     x(1,:) = cresceram(i,2:d+1);
14     ajuda(1,:) = cresceram(i,:);
15     for j = 1:tamanho
16         x(j+1,:) = x(1,:) + round(alfa.*(rand(1,d)-0.5));
17         for l = 1:d
18             if x(1+j,l) < linf(1,l)
19                 x(1+j,l) = linf(1,l);
20             else
21                 if x(1+j,l) > lsup(1,l)
22                     x(1+j,l) = lsup(1,l);
23                 end
24             end
25         end
26         ajuda(1+j,2:d+1) = x(j+1,:);
27         ajuda(1+j,1) = fobj(x(1+j,:));
28     end
29     ajuda = sortrows(ajuda,1);
30     cresceram(i,:) = ajuda(1,:);
31 end
32 end

```

B.4 CRIAÇÃO DAS FAMÍLIAS - ETAPA LOCAL

```

1  function [ bancodedados ] = filhoteslocais(cresceram,n,ng,lsup,
2     linf,alfa,fobj)
3  % Função responsável por receber um dado grupo de otimização e
4     retornar uma
5     matriz contendo este grupo e seus descendentes, além da
6     avaliação da função
7     objetivo em cada indivíduo.
8  d = length(lsup);
9  bancodedados = zeros(n,d+1);
10 contador = 0;
11 ajuda = zeros(n,1);
12 indices = fazergrupos(n,ng);
13 for i = 1:ng*n
14     tamanho = indices(i,1);
15     x = zeros(n,d);
16     x(contador+1,:) = cresceram(i,2:d+1);
17     bancodedados(contador+1,:) = cresceram(i,:);
18     for j = 1:tamanho
19         x(contador+1+j,:) = x(contador+1,:) + round(alfa.*(rand
20             (1,d)-0.5));

```

```

17     for l = 1:d
18         if x(contador+1+j,l) < linf(1,l)
19             x(contador+1+j,l) = linf(1,l);
20         else
21             if x(contador+1+j,l) > lsup(1,l)
22                 x(contador+1+j,l) = lsup(1,l);
23             end
24         end
25     end
26     ajuda(contador+1+j,1) = fobj(x(contador+1+j,:));
27     bancodedados(contador+1+j,1) = ajuda(contador+1+j,1);
28     bancodedados(contador+1+j,2:d+1) = x(contador+1+j,:);
29 end
30 contador = contador+tamanho+1;
31 end
32 if contador < n
33     termo = n-contador;
34     for l = 1:termo
35         perturb = bancodedados(contador,2:d+1)+ round(alfa.*(
36             rand(1,d)-0.5));
37         perturb = max(perturb,linf);
38         perturb = min(perturb,lsup);
39         bancodedados(contador+1,2:d+1) = perturb;
40         bancodedados(contador,1) = fobj(bancodedados(contador,:))
41     );
42 end
43 end

```

B.5 TORNEIO

```

1 function [ d ] = torneio( matriz,vencedores,tamanhotorneio )
2 % Torneio: função que recebe como entradas uma dada matriz, um
3 % dado número
4 % de indivíduos que devem ser selecionados e um dado tamanho de
5 % torneio a
6 % ser disputado e retorna um grupo de individuos selecionados
7 [a,b] = size(matriz);
8 d = zeros(vencedores,1);
9 c = zeros(tamanhotorneio,1);
10 i = 1;
11 while i < vencedores+1
12     for j = 1:tamanhotorneio
13         c(j)=ceil(a*rand(1));
14     end
15     e = min(c);
16     verification = ismember(d,e);
17     if sum(verification)==0
18         d(i,1) = e;

```

```

17         i = i+1;
18     else
19     end
20 end
21 end

```

B.6 TORNEIO INVERSO

```

1 function [ d ] = torneioinverso( matriz,perdedores,
    tamanhotorneio )
2 % Torneio: função que recebe como entradas uma dada matriz, um
    dado número
3 % de indivíduos que devem ser selecionados e um dado tamanho de
    torneio a
4 % ser disputado e retorna um grupo de indivíduos selecionados.
5 [a,b] = size(matriz);
6 d = zeros(perdedores,1);
7 c = zeros(tamanhotorneio,1);
8 i = 1;
9 while i < perdedores+1
10     for j = 1:tamanhotorneio
11         c(j)=ceil(a*rand(1));
12     end
13     e = max(c);
14     verification = ismember(d,e);
15     if sum(verification)==0
16         d(i,1) = e;
17         i = i+1;
18     else
19     end
20 end
21 end

```