UNIVERSIDADE FEDERAL DO PARÁ

INSTITUTO DE TECNOLOGIA

PROGRAMA DE PÓS GRADUAÇÃO EM ENGENHARIA CIVIL

Tiago Rodrigues da Silva

DESENVOLVIMENTO DE UM PROGRAMA PARA ANÁLISE NÃO LINEAR GEOMÉTRICA E FÍSICA DE TRELIÇAS PLANAS ATRAVÉS DA FORMULAÇÃO CO-ROTACIONAL

Belém – PA
2015

UNIVERSIDADE FEDERAL DO PARÁ

INSTITUTO DE TECNOLOGIA

PROGRAMA DE PÓS GRADUAÇÃO EM ENGENHARIA CIVIL

Tiago Rodrigues da Silva

DESENVOLVIMENTO DE UM PROGRAMA PARA ANÁLISE NÃO LINEAR GEOMÉTRICA E FÍSICA DE TRELIÇAS PLANAS ATRAVÉS DA FORMULAÇÃO CO-ROTACIONAL

Dissertação submetida à Banca Examinadora aprovada pelo Colegiado do Programa de Pós-Graduação em Engenharia Civil do Instituto de Tecnologia da Universidade Federal do Pará, como requisito para obtenção do Grau de Mestre em Engenharia Civil na Área de Estruturas e Construção Civil, orientada pelo Professor Remo Magalhães de Souza, Ph.D.

Belém – PA
2015

UNIVERSIDADE FEDERAL DO PARÁ

INSTITUTO DE TECNOLOGIA

PROGRAMA DE PÓS GRADUAÇÃO EM ENGENHARIA CIVIL

DESENVOLVIMENTO DE UM PROGRAMA PARA ANÁLISE NÃO LINEAR GEOMÉTRICA E FÍSICA DE TRELIÇAS PLANAS ATRAVÉS DA FORMULAÇÃO CO-ROTACIONAL

AUTOR:

TIAGO RODRIGUES DA SILVA

Dissertação submetida à Banca Examinadora aprovada pelo Colegiado do Programa de Pós-Graduação em Engenharia Civil do Instituto de Tecnologia da Universidade Federal do Pará, como requisito para obtenção do Grau de Mestre em Engenharia Civil na Área de Estruturas e Construção Civil, orientada pelo Prof. Remo Magalhães de Souza, Ph.D.

BANCA EXAMINADORA:

_____	_____
Prof. Remo Magalhães de Souza, Ph.D Instituto de Tecnologia - UFPA Orientador	Prof. Dr. Alcebíades Negrão Macêdo PPGEC – UFPA Membro Interno
_____	_____
Prof., Dr, Sandoval José Rodrigues Junior Instituto de Tecnologia - UFPA Membro Externo	Prof. André Maués Brabo Pereira, Ph.D CT – UFF Membro Externo

RESUMO

Neste trabalho, realizou-se um estudo sobre a análise estrutural não linear física e geométrica de treliças planas com base na formulação co-rotacional. Para aplicação desta análise, elaborou-se um sistema computacional para fins didáticos. O presente trabalho apresenta as hipóteses simplificadoras fundamentais do problema e demonstrando todas as equações que governam o problema. Para considerar o comportamento não linear do material, foram utilizadas equações constitutivas de material hiperelástico, e as equações constitutivas do modelo de Menegotto e Pinto. Na fase de implementação computacional utilizou-se a linguagem de programação MatLab, com conceitos gerais de programação orientada a objeto. Para validar o programa, compararam-se os resultados obtidos através deste sistema com os disponíveis na literatura. Nestes testes de validação e exemplos, comprovou-se a eficácia e aplicabilidade dos algoritmos implementados para a análise, e a sua praticidade quanto ao uso, pois é possível utilizar o sistema computacional para ensino e análise não linear de treliças planas.

Palavras Chave: Análise Não Linear, Co-rotacional, Treliças

ABSTRACT

In this work, we carried out a study on the physical and geometrical non-linear structural analysis of plane trusses in co-rotational formulation. For the purposes of this analysis, it elaborated a computer system for teaching purposes. This paper presents the key simplifying assumptions of the problem and demonstrating all the equations governing the problem. To consider the nonlinear behavior of the material, constitutive equations of hyperelastic material were used, and the constitutive equations of Menegotto and Pinto model. Computational implementation phase we used the Matlab programming language, with general concepts of object-oriented programming. To validate the program, we compared the results obtained through this system to the available literature. In these validation tests and examples, it has proven efficacy and applicability of algorithms implemented in the analysis and its practicality in the use, it is possible to use the computer system for teaching and nonlinear analysis of plane trusses.

Key Words: Non Linear Analisys, Corotational, Trusse

LISTA DE FIGURAS

Capítulo 3

Capítulo 4

Capítulo 5

Capítulo 6

SUMÁRIO

1 INTRODUÇÃO

A análise linear de uma estrutura considera que mesmo depois da aplicação dos esforços, as equações de equilíbrio são satisfeitas considerando-se a configuração geométrica indeformada. Para estruturas civis simples e menos arrojadas, este tipo de consideração é aplicável, pois as deformações são muito pequenas em comparação com as dimensões dos elementos estruturais.

Ao passar dos anos, as exigências na análise estrutural foram aumentando, através de estruturas de edifícios com arquitetura mais arrojada e consideração de situações críticas, exigindo que durante a análise fosse considerada a configuração deformada da estrutura, chamado efeito não linear geométrico. A análise não-linear de estruturas tem avançado nas últimas décadas, com o desenvolvimento de métodos matemáticos mais sofisticados e precisos para resolver problemas que envolvem grandes deslocamentos.

Estes últimos avanços da análise não linear tem caminhado junto com avanços no desenvolvimento das máquinas e sistemas de computação, pois nessas análises, exige-se a execução de uma grande quantidade de cálculos, com manipulação de grandes matrizes e utilização de métodos iterativos para resolução de problemas.

1.1 Objetivos

Este trabalho tem como objetivo principal o desenvolvimento de um sistema computacional didático que modele o comportamento de treliças planas considerando a não-linearidade física e geométrica utilizando-se a formulação co-rotacional.

Os objetivos específicos do trabalho foram:

● estudo dos fundamentos teóricos e das formulações matemáticas que governam o comportamento de treliças considerando a não-linearidade geométrica.

● estudo dos fundamentos teóricos e das formulações matemáticas de não-linearidade física.

● estudo dos princípios gerais da abordagem orientada a objeto;

● utilização dos recursos da linguagem de programação MatLab para desenvolver o sistema computacional;

● comparação dos resultados obtidos pelo sistema com os resultados publicados na literatura e por resultados obtidos por outros programas de análise não linear.

● Disponibilização do programa para uso didático e futuras atualizações deste.

1.2 Justificativas

Atualmente, diversos programas comerciais de análise e cálculo estrutural são capazes de realizar a análise não linear de estruturas. Entretanto, é comum que alguns estudantes e profissionais aprendam a operar estes programas e utilizar as funcionalidades de análise não-linear sem entender os conceitos e fundamentos matemáticos que envolvem este tipo de análise estrutural.

Diversos livros e artigos científicos abordam este assunto e se propõem a ensinar os conceitos e os métodos matemáticos que governam este tipo de análise. Entretanto, entende-se que é fundamental para o aprendizado é o desenvolvimento de um software para compreensão de todas as etapas do processo da análise não-linear, entendendo como é feito os cálculos de todas as variáveis que envolvem esta análise para cada elemento estrutural, como as barras, e como os métodos numéricos são utilizados dentro do programa para resolução de etapas desta análise.

No ensino de análise não-linear geométrica e física de estruturas são abordados vários conceitos de mecânica dos sólidos e métodos numéricos para resolução de problemas matemáticos encontrados durante esta análise. A compreensão do processo de análise não-linear de estruturas é auxiliada com o desenvolvimento de um sistema computacional, pois através desta construção, é possível a visualização da interligação dos elementos matemáticos com fundamentos da mecânica dos sólidos.

A orientação a objeto é uma metodologia para desenvolvimento de sistemas computacionais que vem sendo uma das mais utilizadas na atualidade. Muitas linguagens de programação são criadas para facilitar a aplicação de conceitos de orientação a objeto. Uma das vantagem da utilização desta metodologia está no princípio da reutilização de software, pois os programas criados podem ser facilmente extensíveis com novas funcionalidades, sem precisar de alteração da estrutura do programa principal.

1.3 Escopo do trabalho

O estudo realizado é apresentado seguindo as normas da metodologia científica, e foi dividido em capítulos conforme apresentado abaixo.

No capitulo 2, está a revisão bibliográfica contendo um resumo das principais literaturas relacionadas com o tema e utilizadas na elaboração deste estudo.

No capítulo 3, são apresentados os fundamentos e conceitos da análise não linear geométrica de estruturas através da formulação co-rotacional, onde são demonstrados os

cálculos envolvidos neste tipo de análise, inclusive demonstração dos métodos numéricos neste processo. Os conceitos e fundamentos da análise não linear física também são apresentados, e o material utilizado no sistema, chamado de modelo de Menegotto Pinto, tem suas equações apresentadas.

No capítulo 4, são apresentados alguns fundamentos e princípios da orientação a objeto aplicados ao desenvolvimento de sistema computacionais. Focando em como estes princípios podem facilitar etapas de desenvolvimento de um programa, e como sistemas computacionais criados com estes conceitos podem ser facilmente incrementados.

No capítulo 5, é apresentado o processo de construção do sistema computacional para análise não linear física e geométrica baseado na formulação co-rotacional, utilizando os métodos matemáticos apresentados no capítulo 3 e conceitos e fundamentos de desenvolvimento de software apresentados no capítulo 4.

No capítulo 6, são apresentados testes de validação do programa. Os resultados dos testes são comparados com os resultados conhecidos na literatura e com resultados obtidos por programas de análise não linear.

No capítulo 7, finaliza-se o estudo proposto, retirando-se as principais conclusões e dando sugestões para extender o sistema computacional com novas funcionalidades para fins didáticos.

2 Revisão Bibliográfica

Este capítulo apresenta pesquisa bibliográfica feita sobre a análise considerando a não linearidade física e geométrica baseadas na formulação co-rotacional, e também conceitos e técnicas de boa prática de programação e orientação a objetos no desenvolvimento de sistemas computacionais.

2.1 Não Linearidade Física e Geométrica de Estruturas

As teorias de barras, segundo ARANHA (2003), foram inicialmente estudadas a partir do século XVII. Os primeiros estudos feitos sobre o comportamento geometricamente não linear, com o emprego de material linear elástico, foram realizados já no século XVIII por Bernoulli e por Euler.

Na atualidade, o tratamento de grandes rotações das barras pode ser realizado utilizando-se a formulação co-rotacional, a qual é fundamentada na utilização de uma base, ou sistema auxiliar de coordenadas, em geral formada por nós das extremidades do elemento, que se move e gira com o elemento a medida que este se deforma.

Através desta formulação, é feita a transformação entre os sistemas de coordenadas que descrevem os graus de liberdade de um elemento (transformação entre os sistemas básico e local, ou entre os sistemas básico e global). Utilizando esse procedimento, a formulação do elemento no sistema básico é completamente independente da transformação, isto é, no sistema básico o elemento pode ser formulado como geometricamente linear e a não-linearidade geométrica é introduzida na transformação.

BARON (1971) criou algoritmos e fluxogramas para transformar um programa de análise linear de treliças e cabos, com base no método da rigidez direta, em análise não linear utilizando as equações não lineares com a abordagem da rigidez direta para grandes deslocamentos nos nós e pequenas deformações nos elementos. A matriz de rigidez geométrica da estrutura é derivada somente a partir das considerações geométricas e estáticas de um estado de equilíbrio perturbado.

JAGANNATHAN et al (1975) afirma que a maioria dos métodos existentes até então, de análise de estruturas reticuladas eram ou aproximados ou eram aplicáveis a casos especiais. Portanto foi feita uma investigação para estudar o comportamento não-linear geométrico de estruturas reticuladas. O método baseou-se em uma formulação de elemento finito para grandes deslocamentos, utilizando elementos de barras, mas podendo ser expandida para incluir um comportamento não linear geométrico de elementos mais gerais. A formulação pôde ser usada para analisar estruturas espaciais que podem ser submetidas a grandes

deslocamentos e grandes rotações dos membros. Foram mostrados que os limites superior e inferior não são sempre confiáveis, e que a aproximação pela análise não linear pode ser utilizada para determinar com precisão os esforços de grandes estruturas.

Um teste de movimento de corpo rígido é proposto por YANG et al (1987) para a verificação da legitimidade de um elemento finito para ser utilizado numa análise não linear geométrica. O teste requer que as forças iniciais atuando sobre um elemento que pode girar não se alterem com a imposição de um movimento de corpo rígido, de modo que as magnitudes das forças iniciais permaneçam inalteradas, de modo a preservar o equilíbrio do elemento durante o movimento de corpo rígido. Os autores concluem que este teste é consistente com as leis físicas básicas, e é, portanto, mais racional do que considerações anteriores sobre movimentos de corpo rígido. Este estudo também demonstra a importância da utilização de uma análise não linear iterativa consistente durante o processo de solução.

LEU et al (1990) afirma que de acordo com a lei de movimento de corpo rígido, um corpo inicialmente influenciado por um conjunto de forças em equilíbrio não vai experimentar nenhum esforço adicional quando sujeito a um movimento de corpo rígido, e permanecerá no equilíbrio após o movimento de corpo rígido. Quando tal lei não é estritamente seguida em uma análise não linear iterativa incremental, forças fictícias podem ocorrer em cada etapa intermediária, o que eventualmente pode resultar em desvio do comportamento calculado de uma estrutura. Do ponto de vista da precisão de solução, o efeito de corpo rígido desempenha um papel mais importante do que o efeito de alongamento. Neste papel, ambos estes efeitos são considerados na geração das matrizes de rigidez não-lineares para um elemento de treliça.

Em MARQUES (1990) apresentou uma formulação do tipo incremental iterativa destinada a análise não linear de pórticos espaciais. Considerou-se os efeitos não lineares introduzidos pelas mudanças de configuração geométrica da estrutura e também pela combinação destes efeitos com aqueles inerentes ao comportamento plástico exibido pelo material. As relações cinemáticas empregadas permitem a consideração de deslocamentos arbitrariamente grandes, acompanhadas de pequenas deformações.

ARANHA (2004) também desenvolveu um trabalho que consiste em um estudo numérico sobre o comportamento de estruturas formadas por barras sujeitas a grandes deslocamentos. Para isto, apresentou a formulação de um elemento finito de barra para análise estática e dinâmica geometricamente não-linear de pórticos planos. O tratamento de grandes rotações das barras foi realizado utilizando-se a Formulação Co-rotacional. Foi desenvolvido um programa na plataforma Matlab para validação da metodologia proposta. Através do programa desenvolvido, pode-se realizar análises estática, modal e dinâmica (linear ou não-

linear). Vários exemplos foram apresentados, comparando-se os resultados da formulação proposta com soluções analíticas simplificadas e com resultados de outros modelos apresentados na literatura. Ficou demonstrada a eficiência e precisão da formulação co-rotacional pela facilidade de tratamento de grandes rotações

SAFFARI (2008) conclui que a fim de avaliar o comportamento das estruturas com precisão, um algoritmo matemático adequado e um conjunto de hipóteses adequadas relativas aos comportamentos estruturais devem ser adotados. Quanto mais preciso o algoritmo e as premissas adotadas, mais preciso é o comportamento das estruturas. No caso de análise não linear, o processo de avaliação é complexo, mas de custo eficaz. No método proposto, no caso de análise não-linear, esta análise é feita aplicando métodos de convergência iterativo para obtenção da trajetória de equilíbrio. Neste estudo, as condições de equilíbrio são atendidas utilizando o algoritmo iterativo de Newton-Raphson.

LE (2012) afirma que estruturas flexíveis são populares em engenharia civil e mecânica. Muitas destas estruturas sofrem grandes deslocamentos e rotações finitas, mas com pequenas deformações. Seus comportamentos dinâmicos são normalmente investigados usando elementos finitos de viga. Um método bem conhecido para formular tais elementos de viga é a abordagem co-rotacional. Este método tem sido amplamente utilizado na análise estática não linear. No entanto, a sua aplicação na dinâmica não-linear é bastante limitado.

BOSCO et al (2014) afirma que o modelo proposto por Menegotto e Pinto é amplamente utilizado para simular a resposta cíclica de estruturas metálicas e barras de aço de estruturas de concreto armado. Devido à simplicidade da sua formulação, é aplicado em inúmeros programas destinados à resposta de estruturas, por exemplo, o OpenSees (Open System for Earthquake Engineering Simulations). Este modelo consiste em um material para comportamento uniaxial e é bastante utilizado para a descrição das respostas do aço em seções transversais discretizadas por meio de fibras e sujeitas a tensões normais. A primeira formulação do modelo foi publicada em 1970 por Giuffré e Pinto. As relações analíticas tiveram a sua origem a partir de um estudo de investigação anterior por Goldberg e Richard (1963) e considerou a possibilidade de descarregar e recarregar o material estudado. Pouco tempo depois, o modelo proposto por Giuffré e Pinto foi melhorado por Menegotto e Pinto (1973) para simular endurecimento cinemático. Em 1983, Filippou et al. melhoraram o modelo original, introduzindo endurecimento isotrópico e revelando uma falha na formulação o original. Especificamente, esses pesquisadores observaram que, se um descarregamento parcial ocorre em deformações menores do que o valor máximo, quando recarregada proporciona forças que são maiores do que as esperadas. Não obstante, Filippou et al. (1983)

consideram que esses erros não foram particularmente significativos e não propõem qualquer alteração na formulação do modelo na opinião dos autores. Esta conclusão pode ser questionável e, portanto, BOSCO et al(2014), propõem uma versão melhorada do modelo Menegotto e Pinto.

FAROUGHI (2015) desenvolveu um novo algoritmo estável para a análise não linear geométrica de estruturas de treliça com base no princípio paramétrico variacional e a abordagem co-rotacional. O modelo constitutivo, explicado resumidamente abaixo, foi proposto para a modelagem de estruturas. O princípio paramétrico variacional é introduzido primeiro para análise com pequena deformação, e, em seguida, a formulação não linear geométrica é originada combinando a formulação do princípio paramétrico variacional linear com o sistema co-rotacional. O método de Newton-Raphson é empregado para resolver as equações não lineares resultantes. Tratamento de convergência de diferentes sistemas computacionais para análise não linear geométrica foi também discutido em pormenor, o que demonstrou a boa estabilidade numérica e a eficácia do método proposto.

2.2 Metodologia de Desenvolvimento de Sistemas Computacionais

FUJII (1997) desenvolveu uma ferramenta computacional orientada a objetos a ser utilizada na análise de estruturas tridimensionais. Essa ferramenta computacional é baseada no método dos elementos finitos e é implementada através de conceitos da programação orientada a objetos. A escolha deste tipo de filosofia de programação, deve-se, principalmente, às suas características de extensibilidades e adaptatividade de código computacional. Um programa que possui uma estrutura orientada a objetos permite uma completa reutilização do código. A implementação de novas classes torna-se uma tarefa mais fácil e rápida, consequentemente outros tipos de análise envolvendo problemas dinâmicos ou processamento paralelo podem ser prontamente adicionados. Particularmente, nesta ferramenta descrevem-se as implementações de elementos de vigas e de treliças, elementos de placa e funções para imposição de vinculações internas nodais. Para validar os resultados obtidos pela ferramenta apresentada, foram estudados problemas para os quais soluções analíticas são conhecidas ou soluções numéricas aproximadas são geradas através de programas comerciais que utilizam técnicas convencionais.

VINCENZI (2004) afirma que o desenvolvimento de software baseado no paradigma Orientado a Objetos e baseado em componentes é uma realidade. O trabalho desenvolvido, trata de teste e validação dentro desse contexto. Observa-se que diversos trabalhos relacionados ao teste de programas Orientados a Objeto vêm sendo desenvolvidos.

ZAMBONI et al. (2005) observa que o estudante de engenharia possui, nos dias de hoje, muitos recursos e ferramentas computacionais para auxiliá-lo na confecção de seus projetos. Linguagens de programação estão disponíveis para o desenvolvimento de aplicações científicas de forma rápida, com interfaces gráficas atraentes e de fácil manipulação, em ambientes propícios à orientação a objetos com seu pragmatismo voltado para o reuso.

SMITH (2015) afirma que a programação orientada a objetos é a prática de criar uma arquitetura de software que permite flexibilidade através do design modular. Um programador que está utilizando os recursos de orientação a objeto não é necessariamente aquele codificador mais avançado, mas quem escolhe ser um programador mais estratégico. Programação Orientada a Objeto não é uma linguagem; é uma prática de arquitetura e do processo de concepção dos sistemas computacionais.

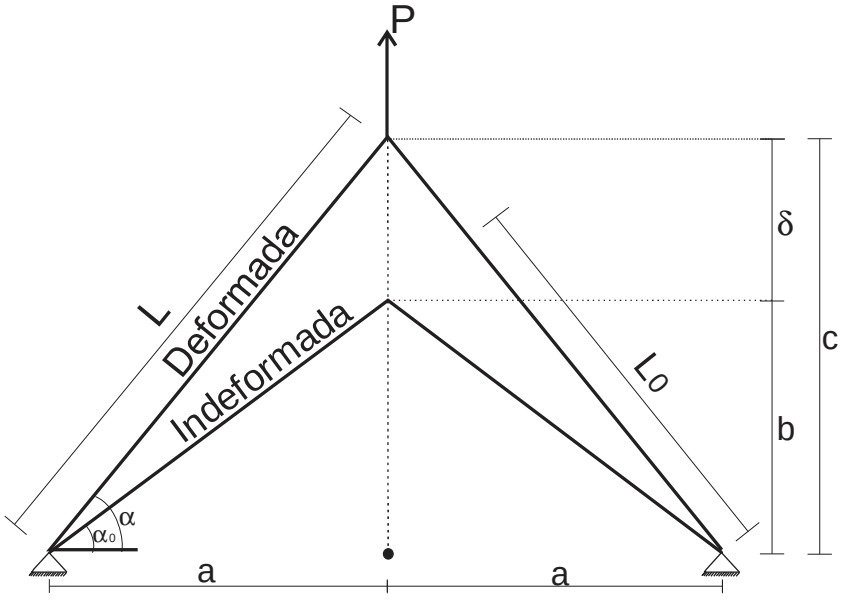
3 ANÁLISE NÃO LINEAR

Neste capítulo é apresentada a formulação utilizada, chamada co-rotacional, para análise não linear geométrica de treliças planas. Também é apresentado o estudo do comportamento não linear do material, e são também mostrados os métodos numéricos que são utilizados em uma análise não linear geométrica e física de treliça. Serão apresentados também a utilização da expansão de uma função em Série de Taylor e o método de Newton Raphson para solução de sistemas não lineares.

3.1 Análise não Linear de um Caso Simples de Treliças

Para melhorar a compreensão do problema da análise não linear geométrica de treliças, nesta seção é apresentada uma solução analítica para um caso simples de treliças como mostrado na Figura 3.1. O problema consiste em uma treliça formada por duas barras e sujeita a uma força vertical no nó central, de modo que a estrutura apresente grandes deformações com um grande deslocamento vertical no ponto de aplicação da força.

Figura 3.1 - Treliça: Geometria indeformada e deformada



De acordo com a Figura 3.1, tem-se as seguintes relações geométricas:

$$\text{sen}(\alpha_0) = \frac{b}{L_0} \qquad \text{Eq (3.1)}$$

$$L_0 = \sqrt{a^2 + b^2} \qquad \text{Eq (3.2)}$$

$$\text{sen}(\alpha) = \frac{b+\delta}{L} \quad \text{Eq (3.3)}$$

$$L = \sqrt{a^2 + (b + \delta)^2} \quad \text{Eq (3.4)}$$

onde L_0 é o comprimento inicial (da configuração indeformada) da barra, L é o comprimento final (da configuração deformada), e δ é o deslocamento vertical do topo da treliça

3.1.1. Equação de equilíbrio (relação estática)

As condições de equilíbrio garantem o equilíbrio estático de qualquer porção isolada da estrutura ou da estrutura como um todo, de acordo com as leis de Newton.

Pode-se obter as seguintes relações geométricas e equações de equilíbrio do problema ilustrado na Figura 3.1. Inicialmente, considera-se o equilíbrio de forças dos nós, na configuração deformada:

$$2\text{Nsen}(\alpha) = P \quad \text{Eq. (3.5)}$$

onde P é força atuando no topo da treliça, e N, é o esforço atuando no eixo da barra.
Substituindo, a Eq. (3.4) na Eq. (3.3), e o resultado na Eq. (3.5), tem-se

$$2N\frac{b+\delta}{\sqrt{a^2+(b^2+\delta^2)}} = P \quad \text{Eq. (3.6)}$$

A relação entre a força e a tensão em uma seção da barra é dada pela relação

$$N = \sigma.A \quad \text{Eq. (3.7)}$$

sendo A, área da seção transversal da barra

Substituindo-se a Eq. 3.7 na Eq. 3.6, chega-se a:

$$2\sigma A\frac{b+\delta}{\sqrt{a^2+(b^2+\delta^2)}} = P \quad \text{Eq. (3.8)}$$

3.1.2. Equações de compatibilidade de deslocamentos (relação cinemática)

A partir das definições de deformação específica ε, e alongamento ΔL tem-se a seguinte relação

$$\varepsilon = \frac{\Delta L}{L_0} = \frac{L-L_0}{L_0} \quad \text{Eq. (3.9)}$$

Substituindo-se a Eq. 3.4 na Eq. 3.9, e considerando ε a deformação da barra, tem-se que

$$\varepsilon = \frac{\sqrt{a^2+(b+\delta)^2}-L_0}{L_0} \qquad \text{Eq. (3.10)}$$

3.1.3. Equação constitutiva

Uma equação constitutiva é uma relação entre as variáveis tensão e deformação. Cada material ou substância tem uma equação constitutiva específica, tal relação só depende da organização molecular interna.

Em mecânica dos sólidos e em engenharia estrutural, as equações constitutivas são igualdades que relacionam as tensões com as deformações em um ponto, De uma forma mais geral, tais equações relacionam componentes dos tensores tensão, deformação e velocidade de deformação. Para um material elástico linear as equações constitutivas se chamam equações de Lamé-Hooke, ou mais simplesmente, lei de Hooke. O comportamento linear-elástico é utilizado nesta demonstração, ou seja, a tensão é proporcional à deformação. Logo, existe uma constante de proporcionalidade entre essas duas grandezas.

$$\sigma = E\epsilon \qquad \text{Eq. (3.11)}$$

onde σ é a tensão, e E é o módulo de elasticidade.

Substituindo as relações da Eq. 3.10 e da Eq. 3.11 na Eq. 3.8, encontra-se a solução para o problema.

$$P(\delta) = 2EA\left(\frac{\sqrt{a^2+(b+\delta)^2}-L_0}{L_0}\right)\frac{(b+\delta)}{\sqrt{a^2+(b+\delta)^2}} \qquad \text{Eq. (3.12)}$$

Nas próximas seções deste capítulo, será apresentado a linearização de funções. E utilizando esse procedimento, pode-se linearizar a Equação (3.12), obtendo-se:

$$P(\delta) = 2EA\frac{b^2}{L_0^2}\delta \qquad \text{Eq. (3.13)}$$

Este resultado linearizado corresponde à solução que se obtém para o problema considerando deformações infinitesimais.

3.2. Aproximação através de Expansão em Série de Taylor

O polinômio de Taylor de grau n de uma função em torno de um ponto fixo, digamos *a*, é o polinômio que interpola a função e suas derivadas até a ordem *n* no ponto *a*. Ou seja, os

valores da função e de suas derivadas, até ordem *n*, no ponto *a* coincide com os valores do polinômio e suas derivadas de ordens correspondentes neste mesmo ponto. É claro que, para que seja possível falar de um polinômio de grau *n* é necessário que a função em questão possa ser derivada pelo menos n vezes no ponto desejado. Pode-se expressar, qualquer função, como um somatório infinito de termos polinomiais. A seguir apresenta-se a dedução da expansão em série de Taylor.

3.2.1. Expansão em torno da origem

Seja uma função *f*(*x*) expressa como um polinômio,

$$f(x) \cong a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots + a_n x^n + \qquad \text{Eq. (3.14)}$$

As derivadas da função *f*(*x*) são:

$$f'(x) \cong 0 + a_1 x^0 + 2a_2 x^1 + 3a_3 x^2 + 4a_4 x^3 + n a_n x^{n-1} + \ldots x^n$$

$$f''(x) \cong 0 + 0 + 2a_2 x^0 + 2.3.a_3 x^1 + 3.4 a_4 x^2 + \ldots + (n-1)(n) a_n x^{n-2}$$

$$f'''(x) \cong 0 + 0 + 0 + 2.3 a_3 + 2.3.4 a_4 x + \ldots + (n-2)(n-1)(n) a_n x^{n-3} + \ldots$$

$$f^i(x) \cong i!\, a_i + \ldots + \frac{n!}{(n-i)!} a_n x^{(n-i)} x^n + \ldots \qquad \text{Eq. (3.15)}$$

Considerando $x = x_0 = 0$, tem-se

$$f(x_0) = f(0) = a_0$$

$$f'(x_0) = f'(0) = a_1 = 1!\, a_1$$

$$f''(x_0) = f''(0) = 2a_2 = 2!\, a_2$$

$$f'''(x_0) = f'''(0) = 6a_3 = 3!\, a_3$$

$$f^{iv}(x_0) = f^{iv}(0) = 24a_4 = 4!\, a_4$$

$$f^i(x_0) = f^i(0) = 1!\, a_1$$

Substituindo as constantes a0, a1, etc, na função polinimial, tem-se que

$$f(x) \cong f(0) + \frac{f^i(0)}{1!} x + \frac{f^{ii}(0)}{2!} x^2 + \frac{f^{iii}(0)}{3!} x^3 + \frac{f^{iv}(0)}{4!} x^4 + \cdots + \frac{f^n(0)}{n!} x^n x^n + \ldots \qquad \text{Eq. (3.16)}$$

3.2.2. Expansão em série em torno de um ponto x_0

Para qualquer $x_0 \neq 0$, pode-se demonstrar que

$$f(x) \cong a_0 + a_1(x - x_0) + a_2(x - x_0)^2 + a_3(x - x_0)^3 + a_4(x - x_0)^4 + \ldots a_n(x - x_0)^n x^n + \qquad \text{Eq. (3.17)}$$

Derivando-se a Eq. (3.17) e considerando $x = 0$, tem-se

$$f(x) \cong f(x_0) + f'(x_0)\left(\frac{x - x_0}{1!}\right) + f''(x_0)\left(\frac{x - x_0}{2!}\right)^2 + \ldots + f^n(x_0)\left(\frac{x - x_0}{n!}\right)^n x^n + \ldots \qquad \text{Eq. (3.18)}$$

A linearização de uma função consiste em expandir a função em série de Taylor é manter apenas os dois primeiros termos da expressão que são termos lineares.

Por exemplo, para linearizar a função $\text{sen}(x)$ em torno da origem $\boldsymbol{x_0 = 0}$, faz-se:

$$\text{sen}(x) = f(0) + f'(0)x + \ ... \qquad \text{Eq (3.19)}$$

$$\text{sen}(x) = \text{sen}(0) + \cos(0)\, x$$

$$\text{sen}(x) = 0 + 1.x$$

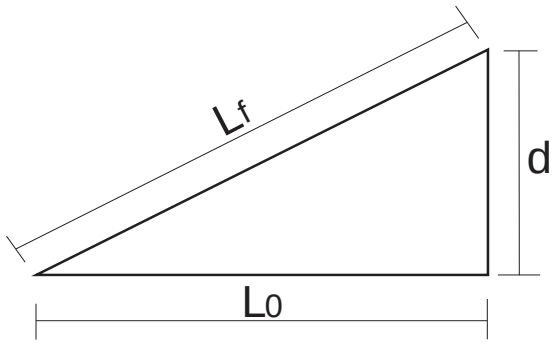
$$\text{sen}(x) = x \qquad \text{Eq (3.20)}$$

3.2.3. Aplicação da série de Taylor

A seguir serão apresentados alguns exemplos com a finalidade de confirmar a aplicabilidade da expansão da série de Taylor em problemas na área de estruturas.

1ª Aplicação: Cálculo do alongamento de uma barra de treliça linear sujeita a um deslocamento imposto na extremidade perpendicular ao eixo inicial da barra

Figura 3.2. – Deformação da barra



Observando a Figura 3.2, pode-se retirar as conclusões abaixo:

$$\Delta L = L_f - L_0 \qquad \text{Eq (3.21)}$$

$$L_f(d) = \sqrt{L_0^2 + d^2} \qquad \text{Eq (3.22)}$$

Substituindo a Eq.3.22 na Eq.3.21, tem-se,

$$\Delta L(d) = L_f - L_0 = \sqrt{L_0^2 + d^2} - L_0 \qquad \text{Eq (3.23)}$$

Observa-se, através da Eq. 3.23 que a função $\Delta L(d)$ tem comportamento não-linear em relação a variável d

Assim, pode-se utilizar a expansão em série de Taylor da função $\Delta L(d)$ para linearizá-la.

Substituindo-se d por x, e ΔL por f, tem-se

$$f(x) = \sqrt{L_0^2 + x^2} - L_0 \qquad \text{Eq (3.24)}$$

Colocando-se a Eq. (3.24) em forma de potência, tem-se

$$f(x) = (L_0^2 + x^2)^{1/2} - L_0 \quad \text{Eq (3.25)}$$

Derivando a Eq. 3.25, tem-se

$$f'(x) = 2x\frac{1}{2}(L_0^2 + x^2)^{-1/2} \quad \text{Eq (3.26)}$$

Considerando $x_0 = 0$, e substituindo nas Eq. 3.25 e Eq. 3.26, tem-se

$$f(x_0) = f(0) = (L_0^2 + 0^2)^{1/2} - L_0 = 0$$

$$f'(x_0) = 2.0.\frac{1}{2}(L_0^2 + 0^2)^{-1/2} = 0$$

Substituindo os resultados acima na forma linearizada de uma função genérica através da expansão em série de Taylor, tem-se a Eq. 3.27

$$f(x) = f(x_0) + f'(x_0)(x - x_0) \quad \text{Eq. (3.27)}$$

$$f(x) = 0 + 0.(x - 0)$$

$$f(x) = 0 \quad \text{Eq. (3.28)}$$

Logo

$$\Delta L(d) = 0 \quad \text{Eq. (3.29)}$$

Conclui-se que o alongamento de uma barra de treliça com comportamento linear sujeita a um deslocamento imposto na extremidade perpendicular ao eixo inicial da barra calculado através da linearização pela série de Taylor é nulo.

2ª Aplicação: Linearização a equação de deformação do problema de treliças com duas barras dado acima.

Utilizando-se a Eq. 3.10

$$\varepsilon(\delta) = \frac{\sqrt{a^2+(b+\delta)^2} - L_0}{L_0} \quad \text{Eq (3.10)}$$

Para o caso de $\delta = 0$, tem-se

$$\varepsilon(0) = \frac{\sqrt{a^2+b^2} - L_0}{L_0} \quad \text{Eq. (3.30)}$$

$$\varepsilon(0) = \frac{L_0 - L_0}{L_0} = 0 \quad \text{Eq. (3.31)}$$

Derivando a Eq. 3.10 em relação a δ, tem-se

$$\varepsilon'(\delta) = \frac{\frac{1}{2}(a^2 + (b+\delta)^2)^{-\frac{1}{2}}.2(b+\delta)}{L_0} \quad \text{Eq. (3.32)}$$

Para o caso de $\delta = 0$, tem-se

$$\varepsilon'(0) = \frac{1}{2L_0}(a^2 + b^2)^{-\frac{1}{2}}.(2b) \quad \text{Eq. (3.33)}$$

$$\varepsilon'(0) = (a^2 + b^2)^{-\frac{1}{2}}.\frac{b}{L_0} \quad \text{Eq. (3.34)}$$

$$\varepsilon'(0) = \frac{b}{L_0.L_0} \quad \text{Eq. (3.35)}$$

$$\varepsilon'(0) = \frac{b}{{L_0}^2} \quad \text{Eq. (3.36)}$$

Substitui-se os resultados acima na forma linearizada de uma função genérica, Eq.3.37, através da expansão em série de Taylor.

$$f(x) = f(x_0) + f'(x_0)(x - x_0) \quad \text{Eq. (3.37)}$$

Para $x_0 = 0$

$$f(x) = f(0) + f'(0).x \quad \text{Eq. (3.38)}$$

$$\varepsilon(\delta) = \varepsilon(0) + \varepsilon'(0)\delta \quad \text{Eq. (3.39)}$$

Ou seja, substituindo-se as Eq.3.31 e Eq.3.36, na Equação 3.40, tem-se

$$\varepsilon(\delta) = 0 + \frac{b.\delta}{L_0^2} \quad \text{Eq. (3.40)}$$

Obtem-se assim como resultado final da linearização da equação de deformação do problema de treliças com duas barras

$$\varepsilon(\delta) = \frac{b.\delta}{L_0^2} \quad \text{Eq. (3.41)}$$

3ª Aplicação: Linearização da Eq.3.12, resultado não-linear do problema de treliças com duas barras dado inicialmente.

$$P(\delta) = 2EA\left(\frac{\sqrt{a^2+(b+\delta)^2}-L_0}{L_0}\right)\frac{(b+\delta)}{\sqrt{a^2+(b+\delta)^2}} \quad \text{Eq. (3.12)}$$

Aplica-se a expansão da série de Taylor em torno da origem, ou seja, considerando $(\delta = x_0 = 0)$

$$f(x) = f(0) + f'(0).x \quad \text{Eq. (3.42)}$$

$$P(\delta) = P(0) + P'(0)\delta \quad \text{Eq. (3.43)}$$

O problema é resolvido encontrando-se os valores dos termos da Eq. 3.43.

Substitui-se o valor $(\delta = 0)$ na Eq. 3.12

$$P(0) = 0 \quad \text{Eq. (3.44)}$$

Deriva-se a Eq.3.12 em relação a δ.

$$P'(0) = \frac{dP}{d\delta} \quad \text{Eq. (3.45)}$$

$$P'(\delta) = \frac{2AE\left[L_0.a^2 - \left(a^2+\delta^2+2b\delta+b^2\right)^{\frac{3}{2}}\right]}{L_0(a^2+b^2+\delta^2+2b\delta)^{\frac{3}{2}}} \quad \text{Eq. (3.46)}$$

Substitui-se o valor $(\delta = 0)$ na Eq. 3.46.

$$P'(0) = \frac{2AE\left[L_0.a^2 - (a^2+b^2)^{\frac{3}{2}}\right]}{L_0(a^2+b^2)^{\frac{3}{2}}} \quad \text{Eq. (3.47)}$$

Utilizando as relações da Eq.3.2, e fazendo operações algébricas, tem-se

$$P'(0) = \frac{2AE[L_0.a^2 - L_0^3]}{L_0^4} \quad \text{Eq. (3.48)}$$

$$P'(0) = \frac{2AE[a^2 - L_0^2]}{L_0^3} \quad \text{Eq. (3.49)}$$

$$P'(0) = \frac{2AEb^2}{L_0^3} \quad \text{Eq. (3.50)}$$

Substituindo os valores e relações obtidas pelas Eq. 3.44 e Eq. 3.50 na Eq. 3.43, obtem-se a solução para o problema de linearização de $P(\delta)$.

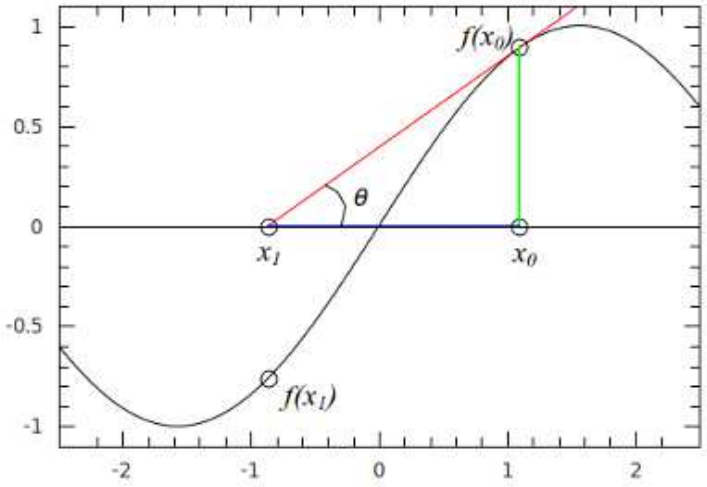
$$P(\delta) = 0 + \frac{2AEb^2}{L_0^3}\delta \quad \text{Eq. (3.51)}$$

$$P(\delta) = \frac{2AEb^2}{L_0^3}\delta \quad \text{Eq. (3.52)}$$

3.2.4. Método de Newton Raphson

O método de Newton-Raphson é um procedimento para encontrar raízes de funções de maneira iterativa. Partindo de um ponto qualquer da função obtém-se o próximo ponto com deslocamento em função do ponto inicial. Faz-se este procedimento repetidamente até encontrar-se o resultado aproximado dentro da tolerância desejada.

Figura 3.3 – Interpretação geométrica do método de Newton Raphson



Pode-se deduzir o método de Newton Raphson a partir da série de Taylor.

$$f(x) \cong f(x_0) + f'(x_0)\left(\frac{x-x_0}{1!}\right) + f''(x_0)\left(\frac{x-x_0}{2!}\right)^2 + \ldots + f^i(x_0)\left(\frac{x-x_0}{i!}\right)^i \quad \text{Eq. (3.53)}$$

$$g(x) = f(x_0) + f'(x_0)(x - x_0) \quad \text{Eq. (3.54)}$$

Considerando $f(x) \approx g(x) = f(x_0) + f'(x_0)(x - x_0)$, e considerando $f(x)$ sendo conhecido, pode-se calcular o valor aproximado de x.

Conhecendo x_0 e $f(x_0)$, pode-se utilizar a série de Taylor

1ª tentativa:

$$x = x_0 + \frac{1}{f'(x_0)}[f(x) - f(x_0)] \qquad \text{Eq. (3.55)}$$

2ª tentativa:

.....Atribui-se o valor de x a x_0 e repete-se até $|x - x_0| \leq tol \; ou \; |f(x) - f(x_0)| \leq tol$.

Este método iterativo de encontrar raízes é usado no sistema computacional para encontrar soluções durante o processo de análise não linear geométrica de treliças.

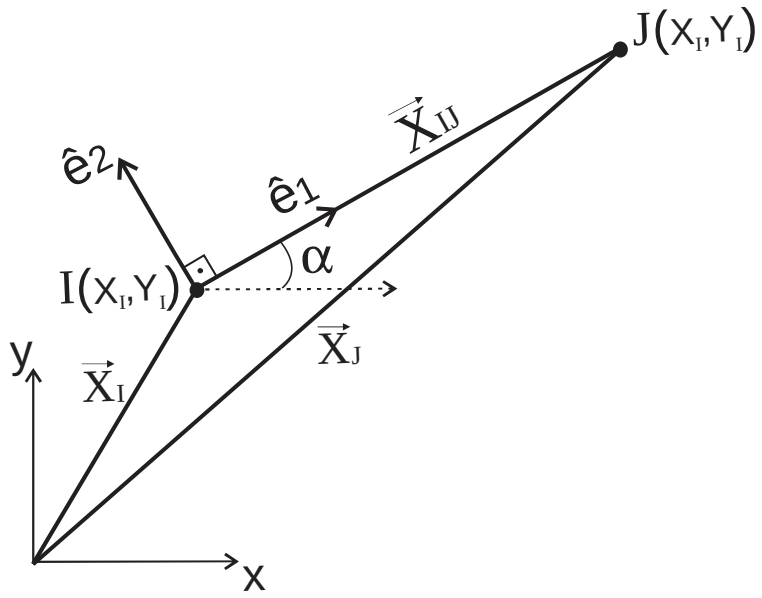
3.3. Formulação Co-rotacional para Análise Não Linear de Treliças Planas.

Na análise utilizando-se a formulação co-rotacional é empregado um sistema de coordenadas sem modos de corpo rígido, denominado sistema básico, mostrado na Figura 3.6. Segundo JUNIOR (2003), a transformação entre os sistemas básico e local, ou entre os sistemas básico e global é feita através da formulação co-rotacional. Utilizando esse procedimento, a formulação do elemento no sistema básico é completamente independente da transformação, isto é, no sistema básico o elemento pode ser formulado como geometricamente linear e a não-linearidade geométrica é introduzida na transformação. Para esse caso, a formulação pode representar arbitrariamente grandes movimentos de corpo rígido, porém com pequenas deformações ao longo do elemento. Ainda assim, se os membros estruturais de um pórtico forem divididos em pequenos elementos, a formulação co-rotacional pode ser utilizada para a solução de problemas com deformações finitas.

3.3.1. Formulação do problema para uma barra da treliça

Uma barra de treliça representada na Figura 3.4, é utilizada para demonstração, e em função das coordenadas dos seus nós I e J.

Figura 3.4 – Deslocamento de uma barra



Observando a Figura 3.4 pode-se jdefinir os vetores de coordenadas dos nós:

$$\overrightarrow{X_i} = \begin{Bmatrix} X_i \\ Y_i \end{Bmatrix} \quad \text{Eq. (3.56)}$$

$$\overrightarrow{X_j} = \begin{Bmatrix} X_j \\ Y_j \end{Bmatrix} \quad \text{Eq. (3.57)}$$

Utilizando-se operações vetoriais, é possível calcular as coordenadas do vetor $\overrightarrow{X_{ij}}$.

$$\overrightarrow{X_i} + \overrightarrow{X_{ij}} = \overrightarrow{X_j} \quad \text{Eq. (3.58)}$$

$$\overrightarrow{X_{ij}} = \overrightarrow{X_j} - \overrightarrow{X_i} \quad \text{Eq. (3.59)}$$

Substituindo-se os valores das Eq. 3.56 e Eq. 3.57, na Eq. 3.59, tem-se

$$\overrightarrow{X_{ij}} = \begin{Bmatrix} X_j - X_i \\ Y_j - Y_i \end{Bmatrix} \quad \text{Eq. (3.60)}$$

$$\overrightarrow{X_{ij}} = \begin{Bmatrix} \Delta_x \\ \Delta_y \end{Bmatrix} \quad \text{Eq. (3.61)}$$

O módulo do vetor X_{ij} é obtido fazendo-se:

$$\left|x_{ij}\right|^2 = \overrightarrow{X_{ij}} . \overrightarrow{X_{ij}} \quad \text{Eq. (3.62)}$$

O cálculo vetorial tem como umas das propriedades que o módulo de um vetor é igual ao seu comprimento. Portanto

$$\ell^2 = \left|x_{ij}\right|^2 \quad \text{Eq. (3.63)}$$

Utilizando-se as propriedades de multiplicações entre vetores, tem-se

$$\ell^2 = \{\Delta x \quad \Delta y\} . \begin{Bmatrix} \Delta x \\ \Delta y \end{Bmatrix} \quad \text{Eq. (3.64)}$$

$$\ell^2 = \Delta x^2 + \Delta y^2 \quad \text{Eq. (3.65)}$$

$$\ell = \sqrt{\Delta x^2 + \Delta y^2} \quad \text{Eq. (3.66)}$$

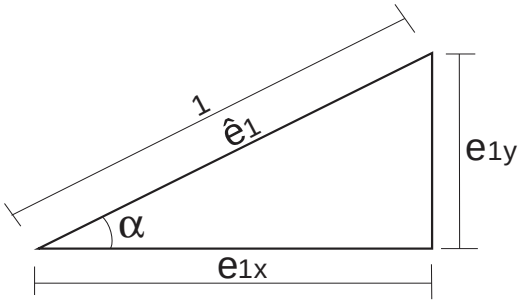
$$\left| x_{ij} \right| = \ell \quad \text{Eq. (3.67)}$$

A determinação dos vetores unitários que formam a base do espaço vetorial do sistema local é calculada através do método abaixo.

$$\hat{e}_1 = \frac{\overrightarrow{X_{ij}}}{|\overrightarrow{X_{ij}}|} \quad \text{Eq. (3.68)}$$

Observando a Figura 3.5, pode ser feita a determinação dos componentes de $\hat{e}_1$ em função de α, onde $\propto$ é o ângulo formado pelos eixos x e x^l.

Figura 3.5 – Componentes do vetor unitário $\hat{e}_1$



O vetor $\hat{e}_1$ é formado pelas seguintes componentes

$$\hat{e}_1 = \begin{Bmatrix} e_{1x} \\ e_{1y} \end{Bmatrix} \quad \text{Eq. (3.69)}$$

Utilizando-se a informações trigonométricas obtidas na Figura 3.5, tem-se

$$\hat{e}_1 = \begin{Bmatrix} \cos \propto \\ \text{sen} \propto \end{Bmatrix} \quad \text{Eq. (2.70)}$$

As componentes do vetor unitário $\hat{e}_2$ são obtidos a partir de $\hat{e}_1$ conforme desenvolvimento abaixo

$$\hat{e}_2 = \begin{Bmatrix} e_{2x} \\ e_{2y} \end{Bmatrix} \quad \text{Eq. (2.71)}$$

Utiliza-se a matriz de rotação 90°, Eq.3.72, denominada R, para o cálculo dos componentes do vetor unitário $\hat{e}_2$ em função das relações trigonométricas.

$$R = \begin{bmatrix} \cos 90^\circ & -\text{sen}\, 90^\circ \\ \text{sen}\, 90^\circ & \cos 90^\circ \end{bmatrix} \quad \text{Eq. (3.72)}$$

$$\begin{Bmatrix} e_{2x} \\ e_{2y} \end{Bmatrix} = \begin{bmatrix} \cos 90^{\circ} & -\operatorname{sen} 90^{\circ} \\ \operatorname{sen} 90^{\circ} & \cos 90^{\circ} \end{bmatrix} \cdot \begin{Bmatrix} e_{1x} \\ e_{1y} \end{Bmatrix} \quad \text{Eq. (3.73)}$$

$$\begin{Bmatrix} e_{2x} \\ e_{2y} \end{Bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \cdot \begin{Bmatrix} e_{1x} \\ e_{1y} \end{Bmatrix} \quad \text{Eq. (3.74)}$$

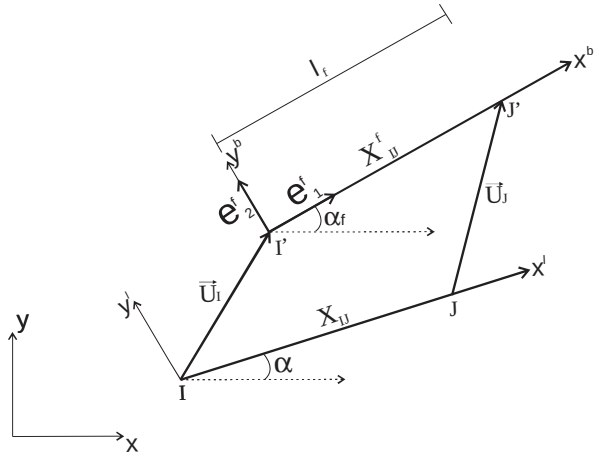
$$\begin{Bmatrix} e_{2x} \\ e_{2y} \end{Bmatrix} = \begin{Bmatrix} -e_1 y \\ e_1 x \end{Bmatrix} \quad \text{Eq. (3.75)}$$

$$\begin{Bmatrix} e_{2x} \\ e_{2y} \end{Bmatrix} = \begin{Bmatrix} -\operatorname{sen} \propto \\ \cos \propto \end{Bmatrix} \quad \text{Eq. (3.76)}$$

$$\hat{e}_2 = \begin{Bmatrix} -\operatorname{sen} \propto \\ \cos \propto \end{Bmatrix} \quad \text{Eq. (3.77)}$$

A Figura 3.6 apresenta a orientação através de diversos vetores da barra em relação aos sistemas global, local e básico.

Figura 3.6 – Sistemas global, local e básico



Na figura 3.6, $\overrightarrow{X_\iota}$ é o vetor posição do nó I, $\overrightarrow{X_J}$ é o vetor posição do nó J, $\overrightarrow{U_\iota}$ é o vetor deslocamento do nó I, e $\overrightarrow{U_J}$o vetor deslocamento no nó J.

Pode-se estabelecer as seguintes relações através da álgebra vetorial

$$\overrightarrow{X_\iota} + \overrightarrow{U_\iota} + \overrightarrow{X^f{}_{\iota J}} = \overrightarrow{X_J} + \overrightarrow{U_\iota} \quad \text{Eq. (3.78)}$$

Isolando o termo $\overrightarrow{X^f{}_{\iota J}}$ na Eq. 3.78, tem-se

$$\overrightarrow{X^f_{\iota J}} = \overrightarrow{X_J} - \overrightarrow{X_\iota} + \overrightarrow{U_J} - \overrightarrow{U_\iota} \quad \text{Eq. (3.79)}$$

Simplificando a Eq. (3.79), tem-se

$$\overrightarrow{X^f_{\iota J}} = \overrightarrow{X_{\iota J}} + \overrightarrow{U_{\iota J}} \quad \text{Eq. (3.80)}$$

O cálculo do comprimento final da barra é feito através do módulo do vetor $\vec{X}^f_{ij}$, conforme o desenvolvimento abaixo

$$\ell^f = \left|\vec{X}_{ij}^f\right| \qquad \text{Eq. (3.81)}$$

A determinação dos vetores unitários que formam a base do espaço vetorial do sistema básico segue de modo análogo a demonstração dos vetores unitários referente ao sistema local, onde $\propto^f$ = ângulo formado pelos eixos x e x^b.

$$\hat{e}_1^f = \frac{\vec{X}_{ij}^f}{\left|\vec{X}_{ij}^f\right|} \qquad \text{Eq. (3.82)}$$

$$\hat{e}_1^f = \begin{Bmatrix}\cos \propto^f \\ \text{sen} \propto^f\end{Bmatrix} \qquad \text{Eq. (3.83)}$$

$$\hat{e}_2^f = \begin{Bmatrix}-\text{sen} \propto^f \\ \cos \propto^f\end{Bmatrix} \qquad \text{Eq. (3.84)}$$

O cálculo do alongamento da barra é feito pela diferença entre o comprimento final e o comprimento inicial.

$$\Delta\ell = \ell^f - \ell \qquad \text{Eq. (3.85)}$$

O processo de subtração dentro de métodos computacionais pode levar à formação de erros nos cálculos numéricos. Para evitar este problema, pode-se usar um artifício para eliminar a operação de subtração no processamento numérico.

O processo inicia-se multiplicando o lado direito da equação Eq. (3.85) pela Eq.(3.86)

$$\left(\frac{(\ell^f + \ell)}{(\ell^f + \ell)}\right) = 1 \qquad \text{Eq.(3.86)}$$

$$\Delta\ell = (\ell^f - \ell)\left(\frac{(\ell^f + \ell)}{(\ell^f + \ell)}\right) \qquad \text{Eq.(3.87)}$$

Utilizando-se propriedades algébricas básicas no tratamento da Eq.(3.87), obtem-se o arranjo da Eq.(3.88)

$$\Delta\ell = \frac{\ell^{f^2} - \ell^2}{\ell^f + \ell} \qquad \text{Eq.(3.88)}$$

Substituindo os valores dos comprimentos finais e iniciais pelos dados vetoriais, no numerador da Eq. (3.88), tem-se o seguinte desenvolvimento.

$$\frac{\ell^{f^2} - \ell^2}{\ell^f + \ell} = \frac{\left(\overrightarrow{X_{IJ}} + \overrightarrow{U_{IJ}}\right)\left(\overrightarrow{X_{IJ}} + \overrightarrow{U_{IJ}}\right) - \left(\overrightarrow{X_{IJ}}\right)\left(\overrightarrow{X_{IJ}}\right)}{\ell^f + \ell}$$

$$\Delta\ell = \frac{\overrightarrow{X_{IJ}}.\overrightarrow{X_{IJ}} + 2\overrightarrow{X_{IJ}}\overrightarrow{U_{IJ}} + \overrightarrow{U_{IJ}}.\overrightarrow{U_{IJ}} - \overrightarrow{X_{IJ}}.\overrightarrow{X_{IJ}}}{\ell^f + \ell}$$

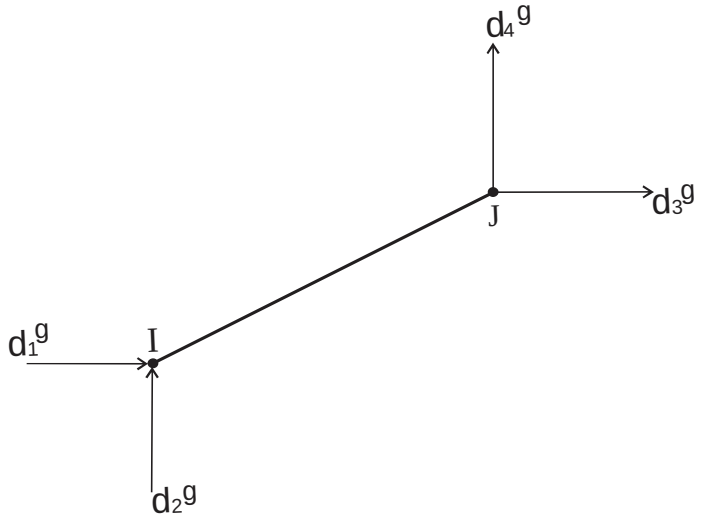
$$\Delta\ell = \frac{2\overrightarrow{X_{IJ}}\overrightarrow{U_{IJ}} + \overrightarrow{U_{IJ}}.\overrightarrow{U_{IJ}}}{\ell^f + \ell}$$

$$\Delta\ell = \frac{\left(2\overrightarrow{X_{IJ}} + \overrightarrow{U_{IJ}}\right).\overrightarrow{U_{IJ}}}{\ell^f + \ell}$$

$$\Delta\ell = \frac{\left(2\overrightarrow{X_{IJ}} + \overrightarrow{U_{IJ}}\right).\overrightarrow{U_{IJ}}}{\ell^f + \ell} \qquad \text{Eq.(3.89)}$$

As componentes do vetor de deslocamentos da barra em relação ao sistema global são representados na Figura 3.7.

Figura 3.7 – Componentes de deslocamento em relação ao sistema global



Assim

$$\{d^g\} = \begin{Bmatrix} d_1^g \\ d_2^g \\ d_3^g \\ d_4^g \end{Bmatrix} \qquad \text{Eq.(3.90)}$$

Os deslocamentos nos nós I e J podem ser deduzidos da Figura 3.7.

$$\overrightarrow{U_\iota} = \begin{Bmatrix} d_1^g \\ d_2^g \end{Bmatrix} \qquad \text{Eq.(3.91)}$$

$$\overrightarrow{U_J} = \begin{Bmatrix} d_3^g \\ d_4^g \end{Bmatrix} \qquad \text{Eq.(3.92)}$$

Logo, substituindo as Eq.(3.91) e Eq.(3.92) na Eq. (3.90), tem-se

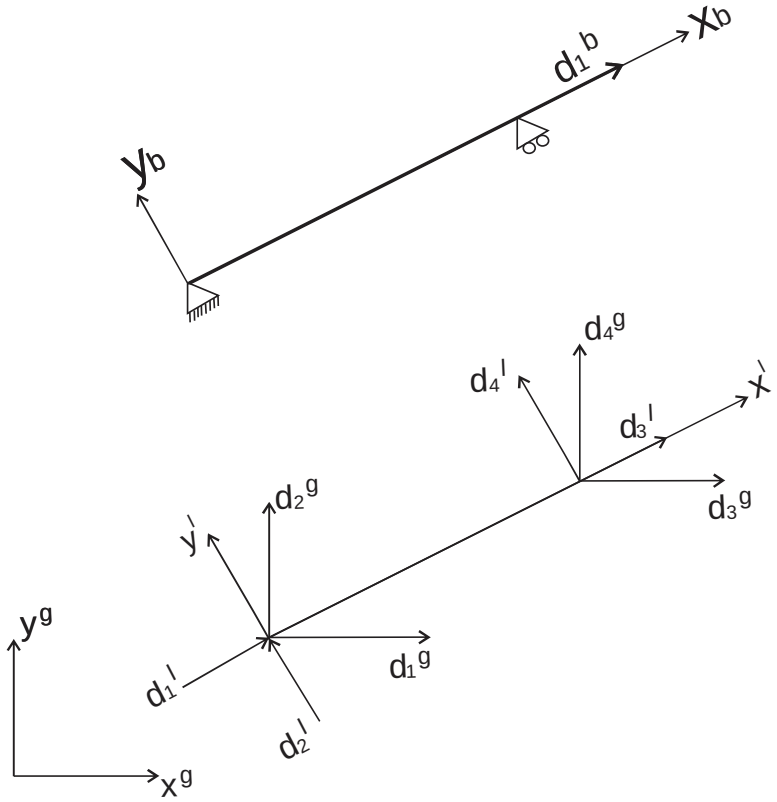
$$\{d^g\} = \begin{Bmatrix} \overrightarrow{U_\iota} \\ \overrightarrow{U_J} \end{Bmatrix} \qquad \text{Eq.(3.93)}$$

Na formulação co-rotacional pode-se utilizar **três sistemas de coordenadas**, a saber:

- Sistema global (x^g, y^g) - Utilizado como referência para toda estrutura.
- Sistema local $\left(x^\ell, y^\ell\right)-$ Utilizado como referência para cada barra individualmente, de modo que o eixo x^ℓ coincide com o eixo da barra na posição **indeformada**.
- Sistema básico $\left(x^b, y^b\right)$ - Utilizado como referência para cada barra individualmente, de modo que o eixo x^b coincide com o eixo da barra na posição **deformada**.

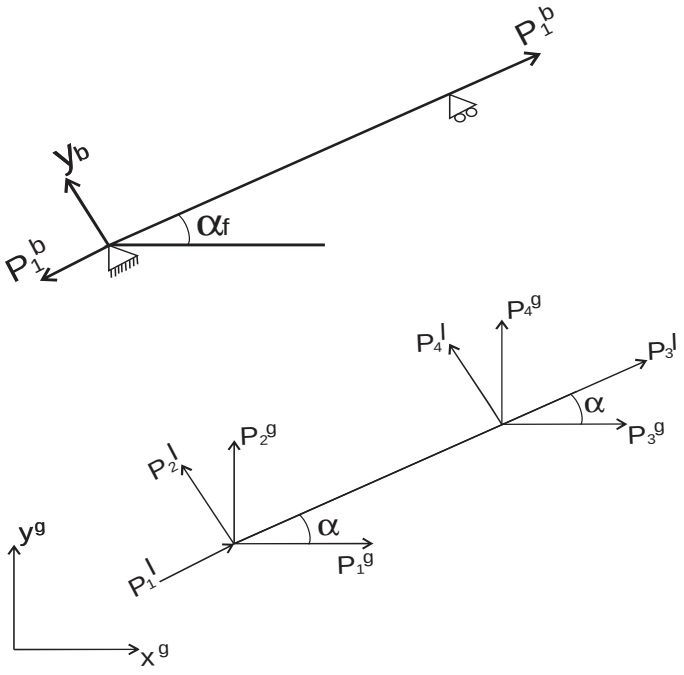
Os deslocamentos da barra nos diversos sistemas (global, local e básico) podem ser calculados e visualizados com a ajuda da Figura 3.8.

Figura 3.8 – Deslocamentos nos sistemas global, local e básico



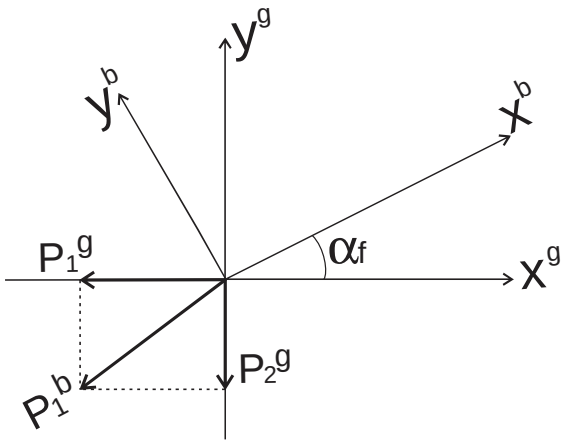
As forças nos diversas sistemas (global, local e básico) podem ser calculados e visualizados com a ajuda da Figura 3.9

Figura 3.9 – Forças nos sistemas globais, locais e básicos



A **relação de equilíbrio** entre as forças do sistema básico e global é apresentada geometricamente nas Figura 3.10 e Figura 3.11.

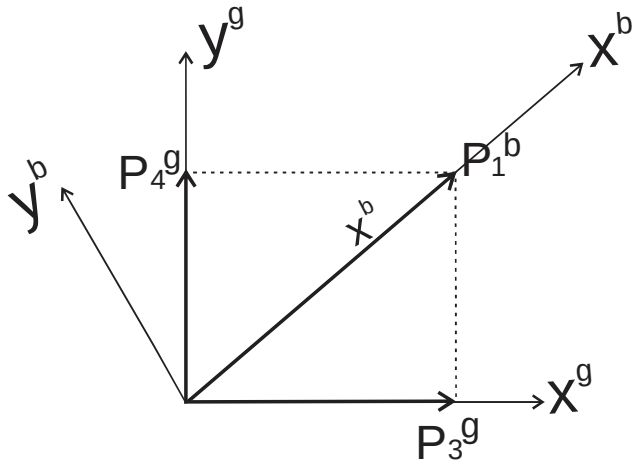
Figura 3.10 – Forças p_1^g e p_2^g nos sistemas global e básico



A análise da Figura 2.10 permite a obtenção das relações apresentadas na Eq. 3.94.

$$\begin{cases} p_1^g = -p_1^b \cos \alpha^f \\ p_2^g = -p_1^b \operatorname{sen} \alpha^f \end{cases} \quad \text{Eq.(3.94)}$$

Figura 3.11 – Forças p_3^g e p_4^g nos sistemas global e básico



A análise da Figura 2.11 permite a obtenção das seguintes relações de equilíbrio.

$$\begin{cases} p_3^g = p_1^b \cos \alpha^f \\ p_4^g = p_1^b \operatorname{sen} \alpha^f \end{cases} \quad \text{Eq.(3.95)}$$

Pode-se rearranjar as Eq. (3.94) e Eq. (3.95), de modo que

$$\begin{cases} p_1^g = -p_1^b \cos\alpha^f \\ p_2^g = -p_1^b \operatorname{sen}\alpha^f \\ p_3^g = p_1^b \cos\alpha^f \\ p_4^g = p_1^b \operatorname{sen}\alpha^f \end{cases} \quad \text{Eq. (3.96)}$$

Utiliza-se propriedades de operações entre matrizes para isolar os termos da Eq.(3.96), resultando em

$$\begin{Bmatrix} p_1^g \\ p_2^g \\ p_3^g \\ p_4^g \end{Bmatrix} = \begin{Bmatrix} -\cos\alpha^f \\ -\operatorname{sen}\alpha^f \\ \cos\alpha^f \\ \operatorname{sen}\alpha^f \end{Bmatrix} . p_1^b \quad \text{Eq.(3.97)}$$

Na forma matricial, tem-se a Eq. (3.98) que representa as relações estáticas entre as forças no sistema global e básico.

$$\{p^g\} = [T]^T p^b \quad \text{Eq. (3.98)}$$

onde, $[T]$ é uma matriz de transformação

$$[T] = \{-\cos\alpha^f \quad -\operatorname{sen}\alpha^f \quad \cos\alpha^f \quad \operatorname{sen}\alpha^f\} \quad \text{Eq. (3.99)}$$

A **relação cinemática**, ou equações de compatibilidade do deslocamento nos sistemas básico e global é desenvolvida abaixo.

No sistema global tem-se

$$d_1^g = \Delta\ell \quad \text{Eq. (3.100)}$$

$$d_1^g = \ell^f - \ell \quad \text{Eq. (3.101)}$$

Os comprimento inicial e final da barra podem ser calculados conforme

$$\ell = \left|\overrightarrow{X_{IJ}}\right| \quad \text{Eq. (3.102)}$$

$$\ell^f = \left|\overrightarrow{X_{IJ}}^f\right| \quad \text{Eq. (3.103)}$$

Utiliza-se as informações vetoriais obtidas através da Figura 3.9 para o cálculo do comprimento final.

$$\ell^f = \left|\overrightarrow{X_{IJ}} + \overrightarrow{U_{IJ}}\right| \quad \text{Eq. (3.104)}$$

$$\ell^{f\,2} = \overrightarrow{X_{IJ}}^f . \overrightarrow{X_{IJ}}^f \quad \text{Eq. (3.105)}$$

$$\ell^f = \left(\overrightarrow{X_{IJ}}^f . \overrightarrow{X_{IJ}}^f\right)^{1/2} \quad \text{Eq. (3.106)}$$

Calcula-se a diferencial do comprimento final, considerando-se

$$\delta y = \frac{dy}{dx}\delta x \quad \text{Eq. (3.107)}$$

Considera-se também a Eq. (3.108)

$$\delta z \;\; = \frac{dz}{dy}.\delta y \qquad \text{Eq. (3.108)}$$

Utilizando-se propriedades da diferenciação, obtém-se a Eq (3.109).

$$\delta z \;\; = \frac{d_z}{d_y}.\frac{d_y}{d_x}\delta x \qquad \text{Eq. (3.109)}$$

Aplicando-se a propriedade da diferenciação da Eq. (3.109) na Eq. (3.106), tem-se

$$\delta \ell^f = \frac{1}{2}\left(\overrightarrow{X_{\iota J}}^f.\overrightarrow{X_{\iota J}}^f\right)^{-1/2}.\delta\left(\overrightarrow{X_{\iota J}}^f.\overrightarrow{X_{\iota J}}^f\right) \qquad \text{Eq. (3.110)}$$

Tratando separadamente o termo da Eq. (3.110), tem-se

$$\delta\left(\overrightarrow{X_{\iota J}}^f.\overrightarrow{X_{\iota J}}^f\right) = \delta\overrightarrow{X_{\iota J}}^f.\overrightarrow{X_{\iota J}}^f + \overrightarrow{X_{\iota J}}^f.\delta\overrightarrow{X_{\iota J}}^f \qquad \text{Eq. (3.111)}$$

Utilizando a propriedade da Eq (3.112) na Eq. (3.111)

$$\vec{a}.\vec{b} = \vec{b}.\vec{a} \qquad \text{Eq. (3.112)}$$

encontra-se

$$\delta\left(\overrightarrow{X_{\iota J}}^f.\overrightarrow{X_{\iota J}}^f\right) = 2\overrightarrow{X_{\iota J}}^f.\delta\overrightarrow{X_{\iota J}}^f \qquad \text{Eq. (3.113)}$$

Observando-se que

$$\left(\overrightarrow{X_{\iota J}}^f.\overrightarrow{X_{\iota J}}^f\right)^{-1/2} = \frac{1}{\ell^f} \qquad \text{Eq. (3.114)}$$

Substitui-se os resultados das Eq. (3.113) e Eq. (3.114) na Eq. (3.110), tem-se:

$$\delta \ell^f = \frac{1}{2}.\frac{1}{\ell^f}.2\overrightarrow{X_{\iota J}}^f.\delta\overrightarrow{X_{\iota J}}^f \qquad \text{Eq. (3.115)}$$

Simplificando-se a Eq. (3.115), tem-se:

$$\delta \ell^f = \frac{1}{\ell^f}.\overrightarrow{X_{\iota J}}^f.\delta\overrightarrow{X_{\iota J}}^f \qquad \text{Eq. (3.116)}$$

Como a relação da pela Eq. (3.117) é verdadeira, tem-se:

$$\overrightarrow{X_{\iota J}}^f = \overrightarrow{X_{\iota J}} + \overrightarrow{U_{\iota J}}, \qquad \text{Eq. (3.117)}$$

Como $\overrightarrow{X_{\iota J}}$ é constante, conclui-se que

$$\delta\overrightarrow{X_{\iota J}}^f = \delta\overrightarrow{U_{\iota J}} \qquad \text{Eq. (3.118)}$$

Logo

$$\delta \ell^f = \frac{1}{\ell^f}.\overrightarrow{X_{\iota J}}^f.\delta\overrightarrow{U_{\iota J}} \qquad \text{Eq. (3.119)}$$

Substituindo a Eq. (3.117) na Eq. (3.119)

$$\delta \ell^f = \frac{1}{\ell^f}.\left(\overrightarrow{X_{\iota J}} + \overrightarrow{U_{\iota J}}\right).\delta\overrightarrow{U_{\iota J}} \qquad \text{Eq. (3.120)}$$

Como

$$\hat{e}_1^f = \frac{\overrightarrow{X_{IJ}}^f}{\ell^f} \quad \text{Eq. (3.121)}$$

$$\hat{e}_1^f = \frac{\overrightarrow{X_{IJ}}+\overrightarrow{U_{IJ}}}{\ell^f} \quad \text{Eq. (3.122)}$$

$$\hat{e}_1^f = \frac{1}{\ell^f}.\left(\overrightarrow{X_{IJ}} + \overrightarrow{U_{IJ}}\right) \quad \text{Eq. (3.123)}$$

Substituindo a Eq. (3.123) em Eq. (3.120), tem-se:

$$\delta\ell^f = \hat{e}_1^f.\delta\overrightarrow{U_{IJ}} \quad \text{Eq. (3.124)}$$

Lembrando-se que

$$\overrightarrow{U_{IJ}} = \overrightarrow{U_J} - \overrightarrow{U_I} \quad \text{Eq. (3.125)}$$

$$\overrightarrow{U_{IJ}} = \begin{Bmatrix} d_3^g \\ d_4^g \end{Bmatrix} - \begin{Bmatrix} d_1^g \\ d_2^g \end{Bmatrix} \quad \text{Eq. (3.126)}$$

$$\delta\overrightarrow{U_{IJ}} = \begin{Bmatrix} \delta d_3^g \\ \delta d_4^g \end{Bmatrix} - \begin{Bmatrix} \delta d_1^g \\ \delta d_2^g \end{Bmatrix} \quad \text{Eq. (3.127)}$$

A relação cinemática, ou equações de compatibilidade do deslocamento no sistema básico é desenvolvida abaixo.

$$d_1^b = \Delta\ell \quad \text{Eq. (3.128)}$$

$$d_1^b = \ell^f - \ell \quad \text{Eq. (3.129)}$$

Deriva-se a Eq. (3.128)

$$\delta d_1^b = \delta\Delta\ell \quad \text{Eq. (3.130)}$$

$$\delta d_1^b = \delta\ell^f - \delta\ell \quad \text{Eq. (3.131)}$$

Como $\delta\ell = 0$, pois ℓ é constante.

$$\delta d_1^b = \delta\ell^f \quad \text{Eq. (3.132)}$$

Utiliza-se a relação obtida através Eq. (3.124), logo

$$\delta d_1^b = \hat{e}_1^f \delta\overrightarrow{U_{IJ}} \quad \text{Eq. (3.133)}$$

Como

$$\hat{e}_1^f = \begin{Bmatrix} \cos\alpha^f \\ \operatorname{sen}\alpha^f \end{Bmatrix} \quad \text{Eq. (3.134)}$$

Substituindo-se a Eq. (3.134) em Eq. (3.133)

$$\delta d_1^b = \langle \cos\alpha^f \operatorname{sen}\alpha^f \rangle \begin{Bmatrix} \delta U_{ij_1} \\ \delta U_{ij_2} \end{Bmatrix} \quad \text{Eq. (3.135)}$$

Logo, substituindo os valores δU_{ij_1} e δU_{ij_2}.

$$\delta d_1^b = \langle \cos\alpha^f \operatorname{sen}\alpha^f \rangle \begin{Bmatrix} \delta d_3^g - \delta d_1^g \\ \delta d_4^g - \delta d_2^g \end{Bmatrix} \quad \text{Eq. (3.136)}$$

Efetuando a multiplicação entre vetores da Eq. (3.136)

$$\delta d_1^b = \langle -\cos\alpha^f \mid -\operatorname{sen}\alpha^f \mid \cos\alpha^f \mid \operatorname{sen}\alpha^f \rangle \begin{Bmatrix} \delta d_1^g \\ \delta d_2^g \\ \delta d_3^g \\ \delta d_4^g \end{Bmatrix} \quad \text{Eq. (3.137)}$$

Sendo $[T]$ a matriz de transformação de deslocamento

$$[T] = \langle -\cos\alpha^f \mid -\operatorname{sen}\alpha^f \mid \cos\alpha^f \mid \operatorname{sen}\alpha^f \rangle \quad \text{Eq. (3.138)}$$

$$\{\delta d^g\} = \begin{Bmatrix} \delta d_1^g \\ \delta d_2^g \\ \delta d_3^g \\ \delta d_4^g \end{Bmatrix} \quad \text{Eq. (3.139)}$$

Por fim, tem-se a relação cinemática (compatibilidade):

$$\delta d^b = [T]\{\delta d^g\} \quad \text{Eq. (3.140)}$$

Lembrando-se da relação estática, ou equação de equilíbrio, dada pela Eq. (3.98)

$$\{\rho^g\} = [T]^T\{\rho^b\}$$

Observa-se na Eq.(3.98) e Eq. (3.140) o princípio da contragradiência generalizada para o caso não linear. Este princípio pode ser deduzido a partir do princípio dos trabalhos virtuais.

Estabelece-se inicialmente a **relação constitutiva**, considerando material linear elástico.

Utiliza-se a Lei de Hooke

$$\sigma = E\varepsilon \quad \text{Eq. (3.141)}$$

O equilíbrio da seção é dado pela relação

$$\sigma = \frac{N}{A_0} \quad \text{Eq. (3.142)}$$

$$\sigma = \frac{P_1^b}{A_0} \quad \text{Eq. (3.143)}$$

A compatibilidade de deformação é dada pela relação Eq. (3.144)

$$\varepsilon = \frac{\Delta\ell}{\ell_0} \quad \text{Eq. (3.144)}$$

Aplicando-se as propriedades constitutivas para seção de área igual a A.

$$P_1^b = \sigma . A \quad \text{Eq. (3.145)}$$

$$P_1^b = EA\varepsilon \quad \text{Eq. (3.146)}$$

$$P_1^b = \frac{EA}{\ell_0}\Delta\ell \qquad \text{Eq. (3.147)}$$

$$P_1^b = \frac{EA}{\ell_0}d_1^b \qquad \text{Eq. (3.148)}$$

Ou seja:

$$p_1^b = \frac{EA}{L}d_1^b \qquad \text{Eq. (3.149)}$$

Estabelece-se a **relação constitutiva** (linear elástico) a nível de barra no sistema básico

$$\{P^b\} = [K^b]\{d^b\} \qquad \text{Eq. (3.150)}$$

onde $[K^b]$ é a matriz de rigidez de barra no sistema básico.

Derivando-se a Eq. (3.150), tem-se

$$\{\delta P^b\} = \delta\Big[[K^b]\{d^b\}\Big] \qquad \text{Eq. (3.151)}$$

$$\{\delta P^b\} = [K^b]\{\delta d^b\} \qquad \text{Eq. (3.152)}$$

Utilizando-se a relação de equilíbrio

$$\{P^g\} = [T]^T\{P^b\} \qquad \text{Eq. (3.153)}$$

Deriva-se a Eq. (3.153)

$$\{\delta P^g\} = \delta([T]^T\{P^b\}) \qquad \text{Eq. (3.154)}$$

Aplicando-se a regra de derivação por partes, tem-se:

$$\{\delta P^g\} = \delta[T]^T\{P^b\} + [T]^T\delta\{P^b\} \qquad \text{Eq. (3.155)}$$

Para resolver esta equação, faz-se

$$\delta(\cos\alpha^f) = (-\operatorname{sen}\alpha^f)\delta\alpha^f \qquad \text{Eq. (3.156)}$$

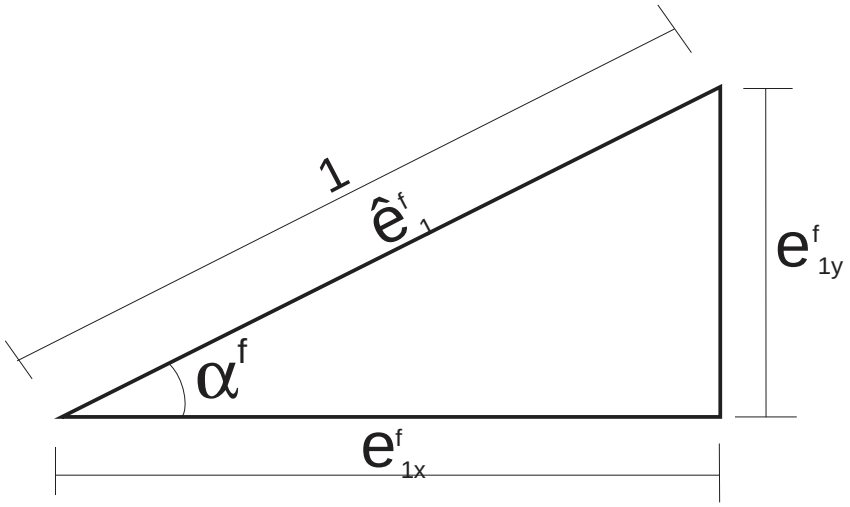
$$\delta(\operatorname{sen}\alpha^f) = (\cos\alpha^f)\delta\alpha^f \qquad \text{Eq. (3.157)}$$

Conclui-se que:

$$[\delta T]^T = \langle \operatorname{sen}\alpha^f \; -\cos\alpha^f \; -\operatorname{sen}\alpha^f \; \cos\alpha^f \rangle^T\delta\alpha^f \qquad \text{Eq. (3.158)}$$

Pode-se extrair as informações sobre as componentes do vetor unitário $\hat{e}_1^f$ através da Figura 3.12.

Figura 3.12 – Componentes do vetor unitário $\hat{e}_1^f$



$$\hat{e}_1^f = \begin{Bmatrix} \cos\alpha^f \\ \text{sen}\,\alpha^f \end{Bmatrix} \quad \text{Eq. (3.159)}$$

$$\hat{e}_2^f = \begin{Bmatrix} -\text{sen}\,\alpha^f \\ \cos\alpha^f \end{Bmatrix} \quad \text{Eq. (3.160)}$$

$$\delta\hat{e}_1^f = \begin{Bmatrix} -\text{sen}\,\alpha^f \\ \cos\alpha^f \end{Bmatrix} \delta\alpha^f \quad \text{Eq. (3.161)}$$

$$\delta\hat{e}_1^f = \hat{e}_2^f \delta\alpha^f \quad \text{Eq. (3.162)}$$

Multiplicando os dois lados por $\{\hat{e}_2^f\}^T$

$$\{\hat{e}_2^f\}^T\{\delta\hat{e}_1^f\} = \{\hat{e}_2^f\}^T\{\hat{e}_2^f\}.\delta\alpha^f \quad \text{Eq. (3.163)}$$

Utilizando-se a propriedade vetorial, conclui-se

$$\{\hat{e}_2^f\}^T\{\hat{e}_2^f\} = 1^2 \quad \text{Eq. (3.164)}$$

Substituindo Eq. (3.164) em Eq. (3.163)

$$\delta\alpha^f = \hat{e}_2^f \delta\hat{e}_1^f \quad \text{Eq. (3.165)}$$

Como sabe-se que

$$\hat{e}_1^f = \frac{\overrightarrow{X_{ij}}^f}{\ell^f} \quad \text{Eq. (3.166)}$$

Resolve-se pela diferencial do quociente

$$\delta\hat{e}_1^f = \frac{\delta\overrightarrow{X_{ij}}^f \ell^f - \overrightarrow{X_{ij}}^f \delta\ell^f}{l_f^2} \quad \text{Eq. (3.167)}$$

Divide-se os termos da Eq. (3.167) para aplicar a regra de derivação

$$\delta\hat{e}_1^f = \frac{\delta\overrightarrow{X_{ij}}^f \ell^f}{\ell^{f2}} - \frac{\overrightarrow{X_{ij}}^f \delta\ell^f}{\ell^{f2}} \quad \text{Eq. (3.168)}$$

Como

$$\overrightarrow{X_{\iota J}}^{f} = \overrightarrow{X_{\iota J}} + \overrightarrow{U_{\iota J}} \quad \text{Eq. (3.169)}$$

$$\delta\overrightarrow{X_{\iota J}} = \delta\overrightarrow{U_{\iota J}} \quad \text{Eq. (3.170)}$$

Substitui-se

$$\delta\ell_1^f = \frac{\delta\overrightarrow{U_{\iota J}}^{f}}{\ell f} - \frac{\overrightarrow{X_{\iota J}}^{f}}{\ell f}.\frac{\delta\ell^f}{\ell f} \quad \text{Eq. (3.171)}$$

$$\frac{\overrightarrow{X_{\iota J}}^{f}}{\ell f} = \hat{e}_1^f \quad \text{Eq. (3.172)}$$

$$\delta e_1^f = \frac{1}{\ell f}\left(\delta\overrightarrow{U_{\iota J}}^{f} - \hat{\mathrm{e}}_1^f.\delta\ell^f\right) \quad \text{Eq. (3.173)}$$

Substituindo os resultados encontrados na Eq. (3.173) em Eq. (2.165).

$$\delta\alpha^f = \hat{e}_2^f\left[\frac{1}{\ell f}\left(\delta\overrightarrow{U_{\iota J}}^{f} - \hat{e}_1^f.\delta\ell^f\right)\right] \quad \text{Eq. (3.174)}$$

$$\delta\alpha^f = \left[\frac{\hat{e}_2^f.\delta\overrightarrow{U_{\iota J}}^{f}}{\ell f} - \left(\hat{e}_2^f.\hat{e}_1^f\right)\delta\ell^f\right] \quad \text{Eq. (3.175)}$$

Como $\hat{e}_2^f$ e $\hat{e}_1^f$ são perpendiculares, tem-se a relação Eq. (3.176)

$$\left(\hat{e}_2^f.\hat{e}_1^f\right) = 0 \quad \text{Eq. (3.176)}$$

Substituindo-se o resultado de Eq. (3.176) em Eq (3.175).

$$\delta\alpha^f = \left[\frac{\hat{e}_2^f.\delta\overrightarrow{U_{\iota J}}^{f}}{\ell f} - (\,0)\delta\ell^f\right] \quad \text{Eq. (3.177)}$$

$$\delta\alpha^f = \frac{\hat{e}_2^f.\delta\overrightarrow{U_{\iota J}}^{f}}{\ell f} \quad \text{Eq. (3.178)}$$

Conhecendo-se os resultados abaixo

$$\begin{Bmatrix}\delta d_3^g & -\delta d_1^g\\ \delta d_4^g & -\delta d_2^g\end{Bmatrix} = \delta\overrightarrow{U_{\iota J}}^{f} \quad \text{Eq. (3.179)}$$

$$(-\operatorname{sen}\alpha^f \cos\alpha^f) = \hat{e}_2^f \quad \text{Eq. (3.180)}$$

Substitui-se as Eq. (3.179) e Eq. (3.180) em Eq. (3.178)

$$\delta\alpha^f = \frac{1}{\ell f}(-\operatorname{sen}\alpha^f \cos\alpha^f)\begin{Bmatrix}\delta d_3^g & -\delta d_1^g\\ \delta d_4^g & -\delta d_2^g\end{Bmatrix} \quad \text{Eq. (3.181)}$$

Efetuando-se a multiplicação em Eq. (3.181), tem-se

$$\delta\alpha^f = \operatorname{sen}\alpha^f.\delta d_1^g - \cos\alpha^f.\delta d_2^g - \operatorname{sen}\alpha^f.\delta d_3^g + \cos\alpha^f.\delta d_4^g \quad \text{Eq. (3.182)}$$

Alterando-se o arranjo da Eq. (3.182)

$$\delta\alpha^f = \frac{1}{\ell f}\langle\operatorname{sen}\alpha^f \,|-\cos\alpha^f|\,-\operatorname{sen}\alpha^f \mid \cos\alpha^f\rangle\begin{Bmatrix}\delta d_1^g\\ \delta d_2^g\\ \delta d_3^g\\ \delta d_4^g\end{Bmatrix} \quad \text{Eq. (3.182)}$$

Onde

$$[S] = \langle \operatorname{sen}\alpha^1 \left| -\cos\alpha^f \right| -\operatorname{sen}\alpha^f \mid \cos\alpha^f \rangle \qquad \text{Eq. (3.183)}$$

$$\delta d^g = \begin{Bmatrix} \delta d_1^g \\ \delta d_2^g \\ \delta d_3^g \\ \delta d_4^g \end{Bmatrix} \qquad \text{Eq. (3.184)}$$

Assim:

$$\delta\alpha^f = \frac{1}{\ell^f}[S]\{\delta d^g\} \qquad \text{Eq. (3.185)}$$

Substituindo-se a Eq.(3.185) em Eq.(2.158), tem-se

$$[\delta T]^T = \begin{Bmatrix} \operatorname{sen}\alpha^f \\ -\cos\alpha^f \\ -\operatorname{sen}\alpha^f \\ \cos\alpha^f \end{Bmatrix} \langle \operatorname{sen}\alpha^f \left| -\cos\alpha^f \right| -\operatorname{sen}\alpha^f \mid \cos\alpha^f \rangle \{\delta d^g\} \frac{1}{\ell^f} \qquad \text{Eq. (2.186)}$$

Onde

$$\underset{\langle \operatorname{sen}\propto^f}{s} \left| \underset{-\cos\propto^f}{-c} \right| \underset{-\operatorname{sen}\propto^f}{-s} \mid \underset{\cos\propto^f \rangle}{c} \qquad \text{Eq. (3.187)}$$

Substituindo as informações dadas em Eq. (3.187) em Eq. (3.186)

$$[\delta T]^T = \frac{1}{\ell^1}\begin{bmatrix} s^2 & -cs & -s^2 & cs \\ -cs & c^2 & cs & -c^2 \\ -s^2 & cs & s^2 & cs \\ cs & -c^2 & -cs & c^2 \end{bmatrix}.[\delta d^g] \qquad \text{Eq. (3.188)}$$

Sendo

$$[G] = \begin{bmatrix} s^2 & -cs & -s^2 & cs \\ -cs & c^2 & cs & -c^2 \\ -s^2 & cs & s^2 & -cs \\ cs & -c^2 & -cs & c^2 \end{bmatrix} \qquad \text{Eq. (3.189)}$$

Colocando a Eq. (3.189) na forma matricial, tem-se

$$[\delta T]^T = \frac{1}{\ell^1}[G].\{\delta d^g\} \qquad \text{Eq. (3.190)}$$

Lembrando-se da Eq. (2.155)

$$\{\delta P^g\} = [\delta T]^T\{P^b\} + [T]^T\delta\{P^b\}$$

Substitui-se em Eq. (3.155) as Eq. (3.190) e Eq. (3.152)

$$\{\delta P^g\} = \frac{1}{\ell^f}[G]P_1^b\{\delta d^g\} + [T]^T[K^b]\{\delta d^b\} \qquad \text{Eq. (3.191)}$$

Lembrando-se da relação de compatibilidade dada na Eq.(2.134)

$$\{\delta d^b\} = [T]\{\delta d^g\}$$

Substitui-se a Eq. (3.140) em Eq. (3.191)

$$\{\delta P^g\} = \frac{1}{\ell f}[G]P_1^b\{\delta d^g\} + [T]^T[K^b][T]\{\delta d^g\} \quad \text{Eq. (3.192)}$$

Isolando o termo $\{\delta d^g\}$ na Eq. (3.192), tem-se

$$\{\delta P^g\} = \left[\frac{1}{\ell f}[G]P_1^b + [T]^T[K^b][T]\right]\{\delta d^g\} \quad \text{Eq. (3.193)}$$

Estabelecendo-se novas variáveis, tem-se

$$[K_G] = \frac{1}{\ell f}[G]\rho_1^b \quad \text{Eq. (3.194)}$$

$$[K_M] = [T]^T[K^b][T] \quad \text{Eq. (3.195)}$$

Onde K_G é a matriz de rigidez geométrica e K_M é uma matriz de rigidez material (com termos geometricamente não lineares). Caso a matriz T fosse linear, a matriz K_M seria uma matriz de rigidez material "pura" (geometricamente linear).

Logo, substituindo-se Eq. (3.194) e Eq. (3.195) na Eq. (3.193), tem-se

$$\{\delta P^g\} = \left[[K_G] + [K_M]\right]\{\delta d^g\} \quad \text{Eq. (3.196)}$$

Sendo$[K^g]$ a matriz de rigidez da barra em relação ao sistema global

$$[K^g] = \left[[K_G] + [K_M]\right] \quad \text{Eq. (3.197)}$$

Substituindo-se a Eq. (3.197) em Eq. (3.196)

$$\{\delta P^g\} = [K^g]\{\delta d^g\} \quad \text{Eq. (3.198)}$$

Se se considerar $\{P^g\}$ uma função em relação a variável $\{d^g\}$

$$\frac{\partial\{P^g\}}{\partial\{d^g\}} = [K^g] \quad \text{Eq. (3.199)}$$

Onde $[K^g]$ é a matriz Jacobiana

Adição da matriz de rigidez da barra na matriz de rigidez da estrutura se faz através do somatório

$$[K] = \sum_{el=1}^{Nel}[H_{el}]^T[K_{el}][H_{el}] \quad \text{Eq. (3.200)}$$

3.3.2. Calcular a trajetória de equilíbrio usando o método de Newton-Raphson

O cálculo da trajetória de equilíbrio usando a formulação co-rotacional apresentada, usando o método de Newton-raphson é demonstrado no algoritmo abaixo.

Algoritmo:

1° passo: Determinar o valor inicial para δ_0. Normalmente em análise estrutural $\delta_0 = 0$.

$$\delta_i = \delta_0$$

2° passo: Avaliar a função no ponto δ_i

$$\rho_i = \rho_{int}(\delta)$$

$$\Delta\rho_i = \rho_{EXT} - \rho_i$$

3° passo: Avaliar a derivada da função em relação a δ no ponto δ_i.

$$k_i = \tan \propto_i = \left.\frac{d\rho_{int}(\delta)}{d\delta}\right|\delta_0$$

4° passo: Determinar variação de $\Delta\delta_i$

$$\Delta\delta_i = (k^{-1})(P_{EXT} - P_i)$$

5°passo: Determinar novo δ_i

$$\delta_{i+t} = \delta_i + \Delta\delta_i$$

$$\delta_i = \delta_{i+1}$$

$$\{D_{i+1}\} = \{D_i\} - ([K]^{-1})(\{P_{EXT}\} - \{P_{INT}(\{Di\})\})$$

6° passo: Avaliar se $\Delta\delta_i$é inferior a tolerância desejada.

Se $(\Delta\delta_i \leq tol)$ for verdadeiro: finalizar processo

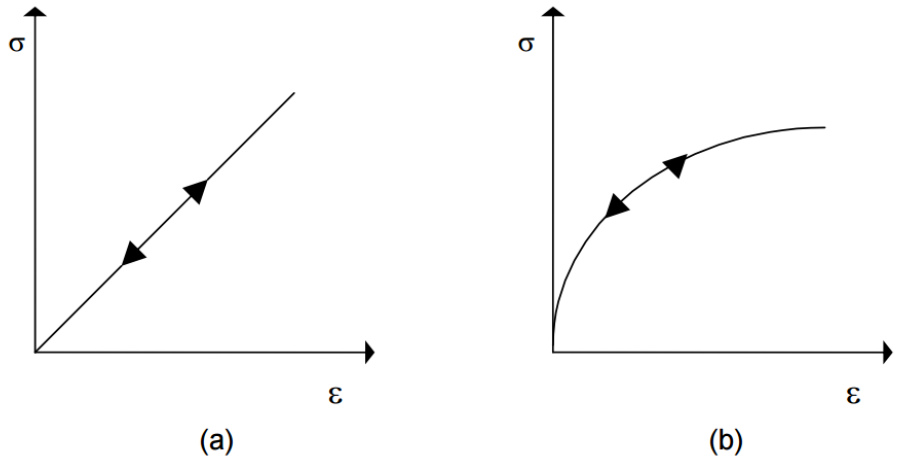
Caso contrário: voltar para o 2° passo até que $\Delta\delta_i seja$ inferior a tolerância.

3.4. Não Linearidade Física

Para se estudar o comportamento tensão-deformação dos materiais é necessário estabelecer teorias sobre os aspectos predominantes deste comportamento. Tais teorias permitem a formulação de equações matemáticas que possibilitam representar o comportamento do material. Estas equações são denominadas equações constitutivas ou leis constitutivas ou modelos constitutivos. Um modelo constitutivo, como tratado neste trabalho, tem como objetivo descrever a relação entre a tensão e a deformação de uma barra quando submetido a variações no seu estado de tensão.

Diz-se que um material é elástico quando a energia dissipada após um ciclo de carregamento-descarregamento é nula e assim, como pode ser visto na Fig. 3.13, as deformações envolvidas no processo são totalmente reversíveis. É importante ressaltar que um material elástico pode apresentar uma curva tensão-deformação linear ou não linear.

Figura 3.13 - Comportamento de material elástico



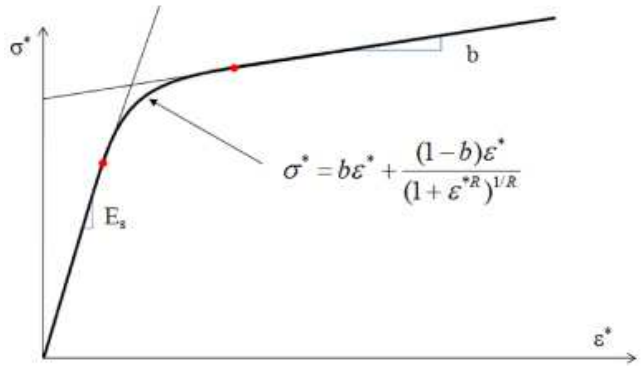
A análise linear é baseada em pressuposições estáticas e de linearidade e, portanto, é válida enquanto essas pressuposições forem válidas. Quando uma (ou mais) dessas pressuposições falham, a análise linear produz previsões erradas e a análise não linear deve ser utilizada como modelo das não linearidades. A pressuposição de linearidade é verdadeira se:

- Todos os materiais do modelo estão de acordo com a Lei de Hook, que afirma que a tensão é diretamente proporcional à deformação. Alguns materiais demonstram esse comportamento apenas se as deformações são pequenas. À medida que as deformações aumentam, as relações tensão-deformação se tornam não lineares. Outros materiais exibem comportamento não linear mesmo com deformações pequenas. Um modelo de material é uma simulação matemática do comportamento de um material. Um material é considerado linear se suas relações tensão-deformação são lineares. A análise linear pode ser usada em modelos com materiais lineares, pressupondo-se que não há outros tipos de não linearidades. Materiais lineares podem ser isotrópicos, ortotrópicos ou anisotrópicos. Sempre que um material no modelo demonstra um comportamento tensão-deformação não linear sob a carga especificada, a análise não linear deve ser usada. A análise não linear pode ser realizada com diversosn tipos de modelos de material.
- Os deslocamento induzidos são pequenos o bastante, portanto, é possível ignorar as alterações na rigidez causadas pelo carregamento. A análise não linear permite representar materiais com relação tensão-deformação bastante diversificados. Os cálculos de matriz de rigidez podem ser refeitos a cada etapa de solução. A frequência do recálculo da matriz de rigidez pode ser controlada pelo usuário.

3.4.1. Modelo de Menegotto Pinto

Neste estudo para simular o comportamento não linear do material, utilizou-se o modelo de Menegotto Pinto.

Figura 3.14 – Curva tensão x deformação do modelo Menegotto Pinto



Este modelo é utilizado para descrever a resposta do aço para ciclos de carregamento e descarregamento. A tensão uniaxial σ é calculada a partir da tensão normalizada σ*. A relação que calcula a tensão normalizada é:

$$\sigma^* = b\varepsilon^* + \frac{(1-b)\varepsilon^*}{(1+\varepsilon^{*R})^{\frac{1}{R}}}$$

Onde ε^* é deformação normalizada, b é taxa de enrijecimento da deformação, e R é um parâmetro que influencia na forma da curva de transição

4. Fundamentos de Desenvolvimento de Sistemas Computacionais

Neste capítulo são apresentados os fundamentos do desenvolvimento de sistemas computacionais, principalmente abordagem orientada a objeto, que direcionaram o desenvolvimento do programa. A compreensão dos conceitos principais apresentados neste capítulo e a utilização correta dos elementos que compõem um sistema, são as condições necessárias para a criação do sistema computacional de análise não linear de treliças.

4.1. Evolução das linguagens de programação

As linguagens de programação têm evoluído (Fig. 3.1) na procura de ferramentas cada vez mais próximas da percepção humana, e que permitam lidar com problemas de maior dimensão e complexidade.

A procura de mecanismos que permitissem a divisão de um problema complexo em sub-problemas começou por dar origem à noção de procedimento. Isso tornou possível raciocinar sobre unidades de menor dimensão.

A interdependência entre as estruturas de dados e as operações que as manipulam implica que se alterada a implementação de uma estrutura de dados tem-se também que alterar as suas operações. E como as unidades do código não eram independentes, não podiam ser compilados separadamente (ALMEIDA; DAROLT, 2001).

O passo seguinte foi a criação de uma abstração que permitisse definir um programa como um conjunto de entidades ou módulos, relativamente autônomos, que encapsulam dados e funcionalidade. Surgiu o conceito de módulo. Segundo DEITEL e DEITEL (2002), um módulo oferece uma interface para o exterior através da qual os seus dados podem ser manipulados. A sua estrutura interna não é conhecida por outras entidades, podendo ser compilado e armazenado em bibliotecas para posterior utilização. Mas não é possível criar dinamicamente várias instâncias de um mesmo módulo.

O conceito de Tipo de Abstração de Dados (TAD) permite a definição de tipos de dados que incluem um conjunto de variáveis e as operações que os manipulam. A partir de um TAD podem ser instanciadas variáveis tal como para qualquer outro tipo de variável (ALMEIDA; DAROLT, 2001). O compilador poderá também verificar a coerência da utilização desse TAD.

Da evolução dos tipos de abstração de dados surge finalmente o conceito de Orientação a Objetos. A idéia foi tornar o modelo anterior mais flexível, permitindo que um tipo seja progressivamente especializado, redefinindo ou incrementando sua funcionalidade. Pretende-se poder reutilizar o software já desenvolvido e testado, adicionando-lhe o comportamento que as novas aplicações necessitem.

Figura 4.1 - Evolução dos conceitos de programação.

Segundo ALMEIDA e DAROLT (2001), um conceito comum a todas as linguagens que se reclamam “Orientadas a Objeto” é o de encapsulamento. O **encapsulamento** é tradicionalmente importante em computação, na medida em que permite decompor grandes sistemas em sub-sistemas autônomos menores que podem ser mais facilmente desenvolvidos e mantidos.

A Abordagem Orientada a Objeto formaliza o encapsulamento, permitindo descrever um sistema como um conjunto de entidades ou objetos autônomos, no sentido de que o funcionamento de um objeto não depende da estrutura interna de outros objetos.

Um outro conceito importante em Orientação a Objeto (OO) é o da **herança**. Através de mecanismos de herança é possível criar novos tipos de objetos partindo dos já existentes através da especificação de como o novo difere do original.

Encapsulamento e herança vão permitir a reutilização das funções de análise não linear de treliças planas já definidas, sem modificação, para resolver os novos problemas.

4.1.1. Encapsulamento

Pode ser definida como sendo o processo de formular conceitos gerais por extração de propriedades comuns de exemplos específicos. Consiste em abstrair uma entidade, através de um conjunto de operações que definem a sua interface externa ou comportamento. As técnicas de abstração de dados, às quais é dada a designação genérica de encapsulamento, têm a ver com dois tipos de mecanismos: modularização e restrições de acessos às informações.

Segundo DEITEL; DEITEL (2002), a modularização consiste na divisão de um sistema em entidades ou módulos, contendo cada uma todas as estruturas de dados e algoritmos necessários para implementar essa parte do sistema (cada módulo pode ser

facilmente reutilizado, e eventuais alterações na funcionalidade de uma entidade não põem em causa a funcionalidade dos componentes restantes).

Restrições de acesso à informação permitem controlar o acesso aos vários elementos de uma entidade, sejam estruturas de dados, ou procedimentos. Os “utilizadores” de um módulo não têm autorização para manipularem os seus detalhes internos, isto é, a única forma de acessar a uma entidade será invocar umas das suas operações externas.

4.1.2. Herança

Em computação os requisitos de um sistema evoluem rapidamente. Segundo ALMEIDA; DAROLT (2001), pode-se considerar evolução como a necessidade de adicionar novas funcionalidades e de modificação das existentes, ou, para sistemas em que os objetivos finais não estejam bem definidos, a evolução pode consistir num desenvolvimento incremental, em que as várias entidades vão sendo sucessivamente especializadas até à solução completa.

Resumindo, a diferença importante da orientação a objeto em relação à programação convencional (imperativa ou procedural) consiste na capacidade de estender um sistema por simples adição de um novo código, em situações onde com as técnicas convencionais, seria necessário modificar o código anterior.

5. Desenvolvimento do Sistema Computacional

Neste capítulo serão mostradas as fases do desenvolvimento do sistema computacional que realiza a análise não linear de treliças utilizando o método co-rotacional. Aplicam-se os conceitos gerais de Programação Orientada a Objeto, apresentadas no capítulo anterior e utiliza-se a linguagem de programação MatLab.

5.1. Processo de Desenvolvimento

Um processo de desenvolvimento de sistemas computacionais é um conjunto de atividades necessárias para transformar exigências de um usuário em um sistema computacional.

Segundo PEREIRA (2004), os processos modernos de desenvolvimento de sistemas computacionais são iterativos e incrementais, repetindo-se sobre uma série de iterações criando o ciclo de vida de um sistema. Cada iteração ocorre sobre um tempo definido e consiste em passar através das exigências, da análise, do projeto, da implementação e atividades de teste, construindo um número de diferentes modelos. Estes modelos estão relacionados uns aos outros e são semanticamente sobrepostos, onde juntos representam o sistema como um todo.

Por outro lado, devido à natureza incremental do processo cada iteração resulta em um incremento nos modelos construídos em iterações precedentes. Um incremento não é necessariamente aditivo. Geralmente, nas fases iniciais do ciclo de vida, um modelo superficial é substituído por outro mais refinado ou sofisticado, porém em fases posteriores os incrementos são tipicamente aditivos, isto é, um modelo é enriquecido com novas características, enquanto as características precedentes são preservadas (PEREIRA, 2004).

O processo utilizado no desenvolvimento do sistema computacional para análise não linear de treliças foi baseado em algumas idéias e fundamentos do processo iterativo e incremental. Para se compreender melhor esse processo de desenvolvimento, faz-se necessário definir alguns termos importantes.

A vida de um sistema computacional pode ser representada como uma série de ciclos. Um ciclo termina com a liberação de uma versão do sistema para os clientes. O ciclo de vida

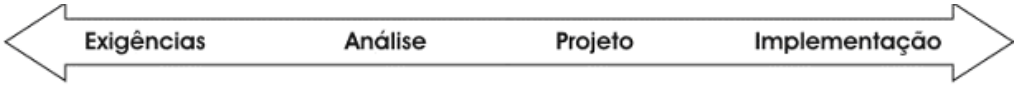
do desenvolvimento de um sistema computacional abrange as fases que um desenvolvedor de sistema vai percorrer para alcançar o resultado desejado (PEREIRA, 2004).

Um ciclo de vida consiste basicamente de cinco fases: planejamento do sistema, que inclui as exigências e investigações iniciais; análise do sistema, que inclui a captura e elucidação das exigências; projeto do sistema; construção e implementação do sistema, que inclui os testes; e distribuição do sistema.

Segundo PEREIRA (2004), uma fase pode ser definida como sendo um intervalo de tempo, onde um conjunto de objetivos bem definidos é alcançado, modelos são desenvolvidos e decisões são tomadas para que se possa avançar para a nova fase. As fases são constituídas por etapas. Uma etapa é definida com sendo um conjunto de atividades bem definidas.

O ciclo de vida de desenvolvimento do sistema computacional foi dividido basicamente em quatro fases e em etapas específicas. Além disso, adotou-se um ciclo no processo de desenvolvimento.

Figura 5.1 - Fases essenciais do processo de desenvolvimento.



5.2. Definição do Problema

Pretende-se desenvolver um sistema computacional orientado a objeto que faça análise não linear de treliças utilizando o método co-rotacional. O programa consiste em três partes. A primeira e mais substancial é o simulador, que possui as equações que modelam o comportamento de treliças utilizando a análise não linear. A segunda é a exibição do modelo na tela, de modo que o usuário possa visualizar a trajetória de equilíbrio, a estrutura deformada e indeformada graficamente. A parte final é a interface gráfica com o usuário, chamado de script, que permite ao usuário controlar a simulação através dos dados de entrada.

O sistema consiste na análise de treliças utilizando a formulação co-rotacional, usando os métodos de cálculo citados nos capítulos anteriores. Onde o usuário entra no arquivo inicial do programa, ou script, com os dados de entrada e nesta simulação são calculados, os esforços que atuam nas barras e nos apoios, e as deformações nos nós.

5.2.1. – Dados de entrada

O usuário deve inserir os dados de entrada necessários para a análise que podem ser divididos em:

Parâmetros geométricos: número de nós e elementos da estrutura, coordenadas dos nós, restrições nodais, cargas nos nós, deslocamentos nodais, conectividade dos elementos e área da seção da barra.

Parâmetros de esforços: cargas e deslocamentos nodais

Parâmetros de análise: número de passos de carga, tolerância e número máximo de iterações para o algoritmo de Newton-Raphson

Parâmetros de material: equação constitutiva do material, limites da curva de comportamento, deformação de interesse.

5.2.2. – Dados de saída

O programa deve retornar os seguintes dados de saída: reações de apoio, tensão e deformação nas barras, deslocamentos nos nós e força nas barras, informações divididas em relação ao eixo horizontal e vertical.

A visualização gráfica do sistema deve mostrar a trajetória de equilíbrio de um ponto através de um gráfico com a relação carga x deslocamento, e a geometria indeformada e deformada da estrutura.

Não será necessário neste sistema computacional um tratamento de exceções, que informe o usuário quando este inserir dados inválidos, ou quando não houver dados de entrada para a realização dos cálculos.

5.3. Análise e Projeto Orientado a Objeto

Denomina-se processo de Análise e Projeto Orientados a Objetos o processo que envolve analisar e projetar um sistema de um ponto de vista orientado a objetos. Esta análise é um termo genérico para as idéias por trás do processo que é empregado para analisar um problema e desenvolver uma abordagem para resolvê-lo. Problemas pequenos não requerem um processo exaustivo. Pode ser suficiente escrever um pseudocódigo antes de se começar a escrever o código. Pseudocódigo é um meio informal de representar o código de um

programa. Não é uma linguagem de programação de verdade, mas pode ser utilizado como um esboço para guiar à medida que se escreve o código (DEITEL; DEITEL, 2002).

Pseudocódigo pode ser suficiente para pequenos programas, mas, à medida que os problemas e os grupos de pessoas que resolvem estes problemas aumentam em tamanho, os métodos da análise e projeto orientado a objeto são mais usados. Nesta fase do processo de desenvolvimento do projeto, é importante definir as funções que irão compor o sistema, e suas distribuições de responsabilidades. Trabalhar com funções aumenta o nível de abstração, pois é possível decompor o sistema em subsistemas independentes.

Para o caso deste programa de análise não linear de treliças, foi necessário criar classes de acordo com as exigências dos cálculos por trás desta análise.

5.3.1. Identificação das Classes do Sistema

Para se identificar os elementos do sistema computacional, adota-se uma sequência padrão de análise e projeto orientado a objeto, que contém três técnicas principais. Segundo PEREIRA (2004), uma proposta de sequência de análise encontra-se descrita a seguir, porém não há uma obrigatoriedade em se seguir essa sequência na mesma ordem que está sendo apresentada.

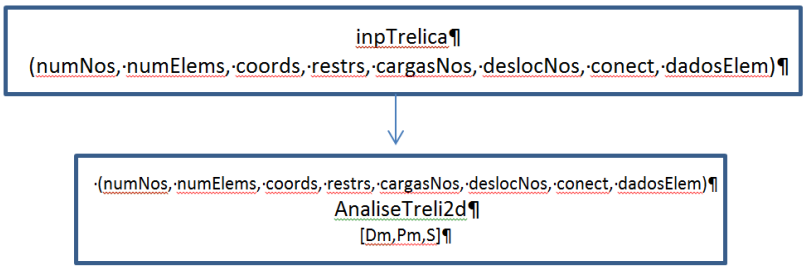
1. Identificação das necessidades do problema real;
2. Identificação dos funções focando nas suas responsabilidades e colaborações no sistema (análise CRC – Classe/Responsabilidade/Colaboração); e
3. Identificação das funções e operações através da extração dos substantivos e verbos através de uma análise textual, a qual terá como base a fundamentação teórica apresentada na descrição do problema.

A escolha da ordem da sequência de análise depende muito das características do modelo que está sendo analisado. A escolha de qual sequência de técnicas de identificação de elementos seria mais adequada para o sistema foi realizada a partir de uma pré-análise, onde se verificou principalmente qual seria a técnica mais adequada como ponto de partida na identificação das funções.

Para o sistema criado, o ponto de partida de análise foi a identificação de objetos modelando-se o problema real. Verificou-se a necessidade de criar um arquivo inicial onde fosse possível inserir todos os dados de entrada. Esse script inicial deve enviar esses dados para a função que fará os cálculos.

A Fig. 5.2, mostra a primeira relação entre as funções previamente identificadas: inpTrelica e AnaliseTreli2d

Figura 5.2 – Diagrama de Classes a partir do Script Inicial



.

Para se identificar as funções restantes, aplicou-se a técnica de cartões CRC. A Fig. 5.3 a Fig. 5.16 apresenta os cartões CRC.

Figura 5.3 - Cartão CRC da função **analiseTrelicasNaoLinear**

analiseTrelicasNaoLinear	
- Realiza os cálculos iniciais e finais da análise não linear da treliças	- Receber todos os dados de entrada para realização dos cálculos. - Retornar os dados de saídas com resultados finais

Figura 5.4 - Cartão CRC da função **calcTensaoGMP**

calcTensaoGMP	
- Realiza o cálculo da tensão e módulo de elasticidade tangente para um material representado pelo modelo de GIUFRE – MENEGOTTO - PINTO	- Receber os dados de entrada contendo informações sobre a curva do comportamento do material e a deformação de interesse. - Retornar os dados contendo a tensão do material e módulo de elasticidade tangente

Figura 5.5 - Cartão CRC da função **compOrientElem**

compOrientElem	
- Calcula o comprimento entre dois pontos e o cosseno e seno diretores da barra.	- Recebe as coordenadas do nós I e J. - Retorna o comprimento da reta entre os nós I e J, e os cossenos e senos formados pela barra.

Figura 5.6 - Cartão CRC da função **desenhaEstrut2d**

desenhaEstrut2d	
-Desenha a estrutura indeformada, com posicionamento das cargas e restrições.	- Recebe o número e coordenadas dos nós, número de elementos e conectividade, cargas e restrições. -Retorna figura com esquema da estrutura incluindo carregamento e restrições nodais.

Figura 5.7 - Cartão CRC da função **desenhaTrelica2dDeformada**

desenhaTrelica2dDeformada	
-Desenha a estrutura deformada.	- Recebe o número e novas coordenadas dos nós, número de elementos e conectividade, cargas e restrições. -Retorna figura com o novo esquema da estrutura incluindo carregamento e restrições nodais.

Figura 5.8 - Cartão CRC da função **ImprimeResultadosTrelica2d**

ImprimeResultadosTrelica2d	
- Gera um arquivo com os resultados da análise.	- Recebe os dados contendo os resultados da análise não linear. - Retorna um arquivo organizado contendo um relatório com os resultados da análise.

Figura 5.9 - Cartão CRC da função **inpTrelica**

inpTrelica	
- Arquivo inicial para inserir todos os dados de entrada para a análise.	- Recebe os dados de entrada e chama as funções de análise, de geração de gráfico, de geração de arquivos de texto com resultados.

Figura 5.10 - Cartão CRC da função **obtemGrausLibElemento**

obtemGrausLibElemento	
- Determina os graus de liberdade do elemento.	- Recebe os dados de entrada com informações dos nós do elemento - Retorna os dados de saída contendo os graus de liberdade do elemento.

Figura 5.11 - Cartão CRC da função **obtemGrausLibEstrut**

obtemGrausLibEstrut	
- Determina os graus de liberdade da estrutura.	- Recebe os dados de entrada com informações sobre os nós e graus de liberdade. - Retorna os dados de saída contendo os graus de liberdade da estrutura, número de graus de liberdade livre da estrutura..

Figura 5.12 - Cartão CRC da função **obtemMatRotacao**

obtemMatRotacao	
- Determina a matriz de rotação.	- Recebe os dados de entrada com informações sobre seno e cosseno da barra em relação ao eixo global. - Retorna os dados de saída contendo a matriz de rotação em relação aos eixos diretrizes.

Figura 5.13 - Cartão CRC da função **obtemVetForcasEMatRigBasico**

obtemVetForcasEMatRigBasico	
- Determina o vetor de forças e a matriz de rigidez tangente para um determinado vetor de deslocamentos em relação ao sistema básico de coordenadas.	- Recebe os dados do material, seção da barra e deformação no sistema básico. - Retorna o vetor de forças e a matriz de rigidez

Figura 5.14 - Cartão CRC da função **obtemVetForcasEMatRigGlobal**

obtemVetForcasEMatRigGlobal	
- Determina o vetor de forças e a matriz de rigidez da barra no sistema global	- Recebe os dados da barra e deformação no sistema global. - Retorna o vetor de forças e a matriz de rigidez da barra no sistema global.

Figura 5.15 - Cartão CRC da função **obtemVetForcasEMatRigEstrutura**

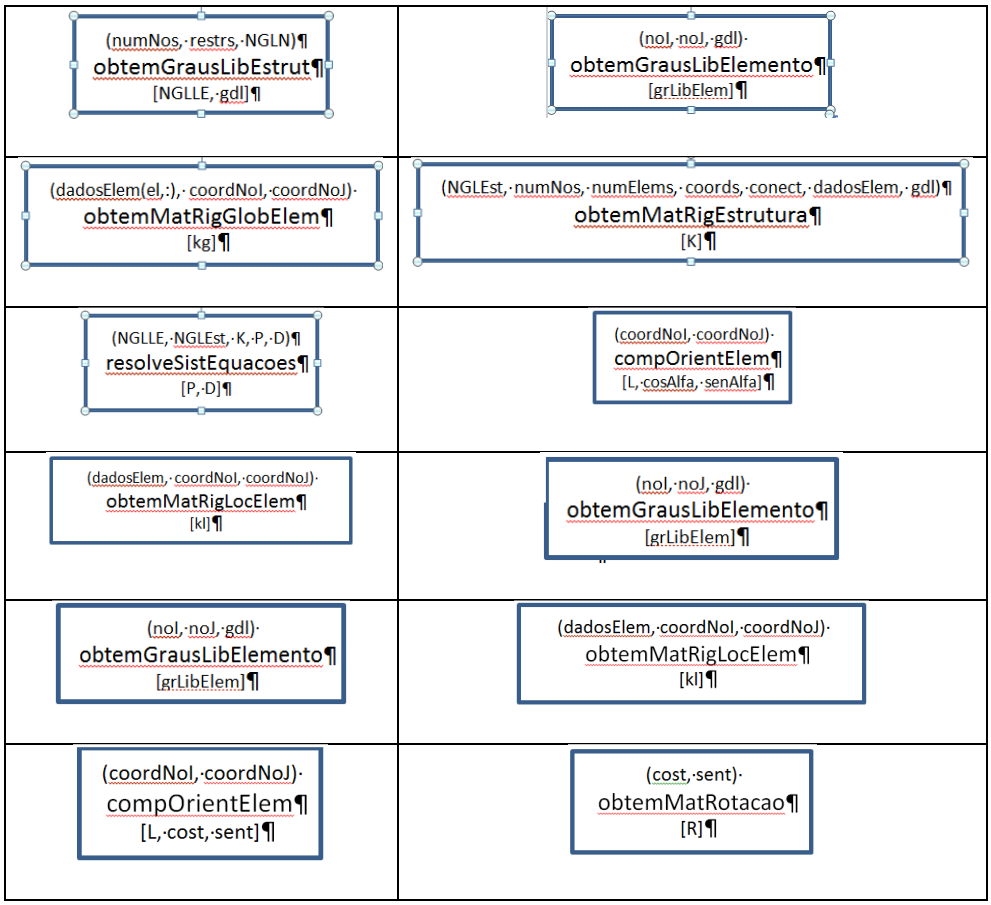
obtemVetForcasEMatRigEstrutura	
- Determina o vetor de forças e a matriz de rigidez da estrutura	- Recebe os dados de cada barra e deformação de todo sistema. - Retorna o vetor de forças e a matriz de rigidez da estrutura

Figura 5.16 - Cartão CRC da função **plotaTrajetóriadeEquilíbrio**

plotaTrajetóriadeEquilíbrio	
- Determina o vetor de forças e a matriz de rigidez da estrutura	- Recebe os dados de cada barra e deformação de todo sistema. - Retorna o vetor de forças e a matriz de rigidez da estrutura

A Figura 5.17 apresenta os dados de entrada e saída de cada função do sistema

Figura 5.17 – Dados de entrada e saída de cada função



Na Figura 5.18, Figura 5.19 e Figura 5.20 observa-se as principais relações entre as classes mostradas anteriormente através de fluxogramas que revelam toda a trajetória dos dados de entrada e saída durante a análise.

Figura 5.18 – Fluxograma geral das funções durante a análise

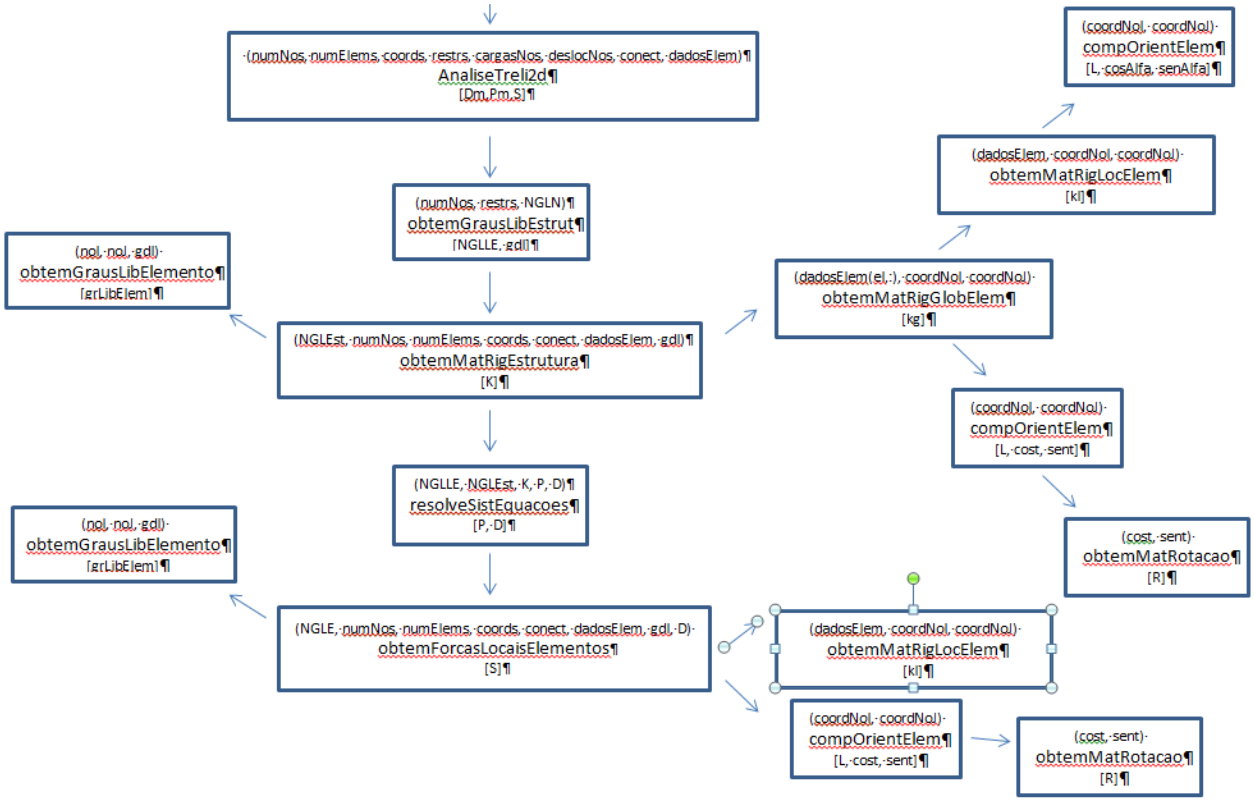


Figura 5.19 – Fluxograma a partir da função obtemMatRigGlobElem

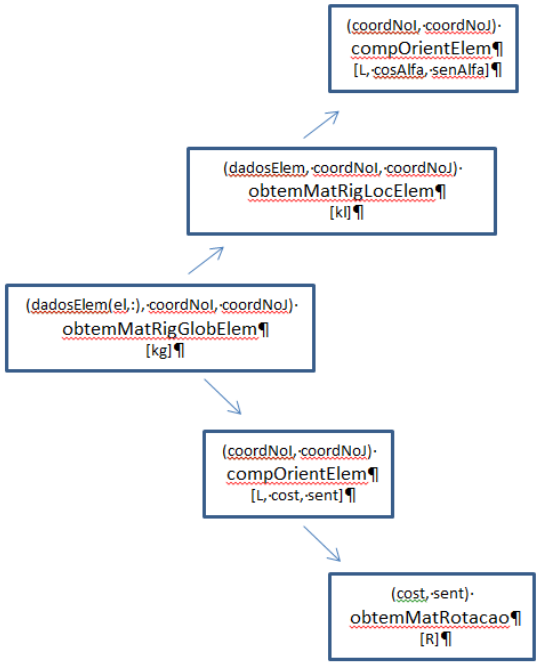
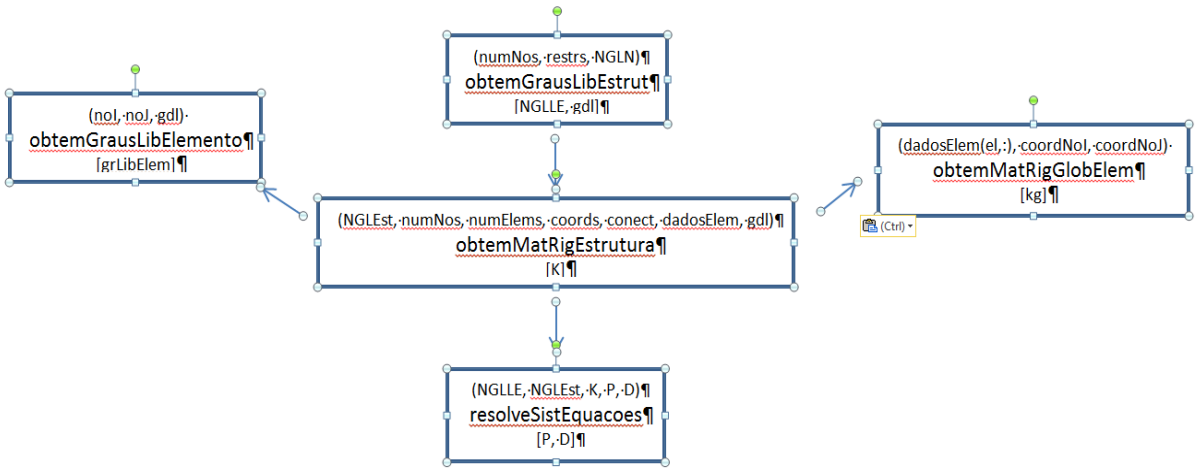


Figura 5.20 – Fluxograma a partir da função obtemMatRigEstrutura



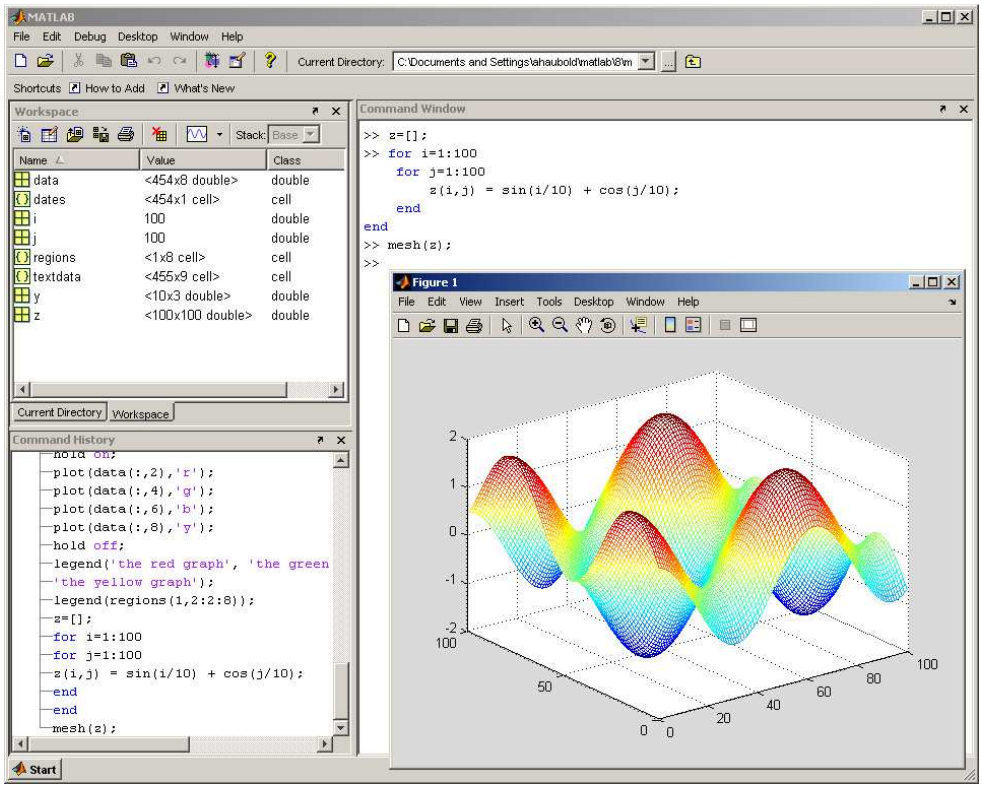
5.3.2. Ambiente de Desenvolvimento

Para o desenvolvimento dos códigos das classes utilizou-se MatLab (MATrix LABoratory), trata-se de um software interativo de alta performance voltado para o cálculo numérico. O MATLAB integra análise numérica, cálculo com matrizes, processamento de sinais e construção de gráficos em ambiente fácil de usar, onde problemas e soluções são expressos como eles são escritos matematicamente, ao contrário da programação tradicional. Uma das grandes vantagens em se utilizar o MatLab é facilidade para tratamento de problemas da matemática.

O MATLAB é um sistema interativo cujo elemento básico de informação é uma matriz que não requer dimensionamento. Esse sistema permite a resolução de muitos problemas numéricos em apenas uma fração do tempo que se gastaria para escrever um programa semelhante em linguagem Fortran, Basic ou C.

O MATLAB possui uma interface (Fig. 5.21) relativamente simples e possibilita o desenvolvimento de aplicativos com um editor de código de uso bastante fácil que pode ser tanto por via textual como gráfica. Além disso, o MatLab disponibiliza um ambiente próprio de compilação, execução e depuração de erros.

Figura 5.21 –Ambiente de Desenvolvimento - MatLab



.

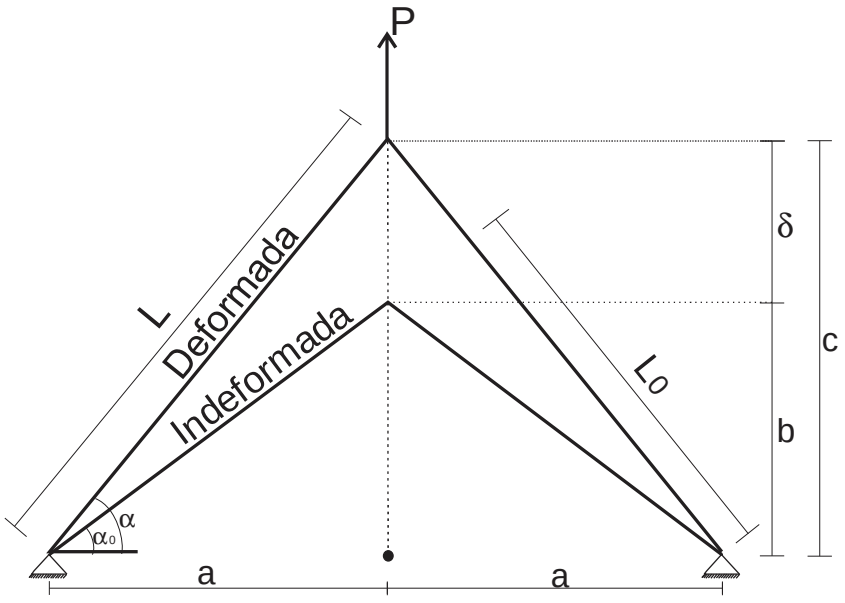
6. TESTES DE VALIDAÇÃO

Neste capítulo é apresentado o funcionamento geral do sistema computacional orientado a objetos que faz análise de treliças planas considerando a não linearidade geométrica e física, cujo desenvolvimento encontra-se descrito no Capítulo 4. O funcionamento do programa será descrito baseando-se em um teste de validação produzido para avaliar a eficiência e a desempenho dos algoritmos propostos neste trabalho. Neste teste, os resultados provenientes das equações implementadas são comparados com exemplos retirados da bibliografia consultada.

6.1. Teste de validação 1

O primeiro teste de validação é a comparação entre o resultado obtido através do programa e o resultado obtido através dos cálculos analíticos da estrutura representada na Figura 6.1.

Figura 6.1. Treliça – Geometria indeformada e deformada



A estrutura representada esquematicamente na Figura 6.1, corresponde a uma treliça com duas barras simétricas de material linear elástico, com apoio de segundo gênero em uma extremidade de ambas, e uma carga na direção vertical na outra extremidade. Sendo a = 2m, b = 1,5m, área da seção das barras é igual a $5x10^{-4}$ m^2 e Módulo de elasticidade para ambas as barras é igual a $2x10^{8}$MPa.

O resultado analítico é obtido através da equação 3.12, mostrada novamente abaixo. Esta equação foi demonstrada no capítulo 3.

$$P(\delta) = 2EA\left(\frac{\sqrt{a^2 + (b + \delta)^2} - L_0}{L_0}\right)\frac{(b+\delta)}{\sqrt{a^2 + (b + \delta)^2}}$$

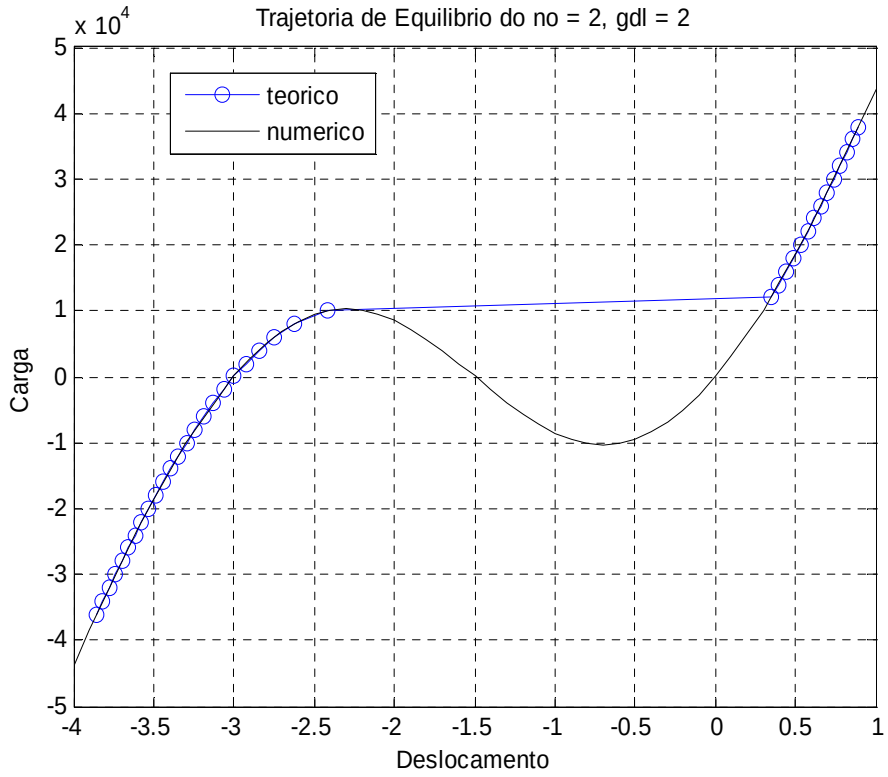
6.1.1. Teste de validação 1 – Parte 1

Para obter-se o resultado pelo programa, alterou-se o arquivo de entrada de dados com informações sobre material, dimensões dos elementos e seções, e carregamento aplicado, conforme mostrado no Anexo B.1.

No primeiro teste, o programa foi preparado para aplicar, primeiramente, uma carga no sentido para baixo de -4,5x10^{4}kN, e variar o carregamento no sentido para cima com o valor de 4,5x10^{4}kN.

A Figura 6.2 compara a trajetória de equilíbrio da estrutura calculada através dos resultados obtidos de forma analítica (linha de cor preta) com os dados numéricos (linha com círculos de cor azul).

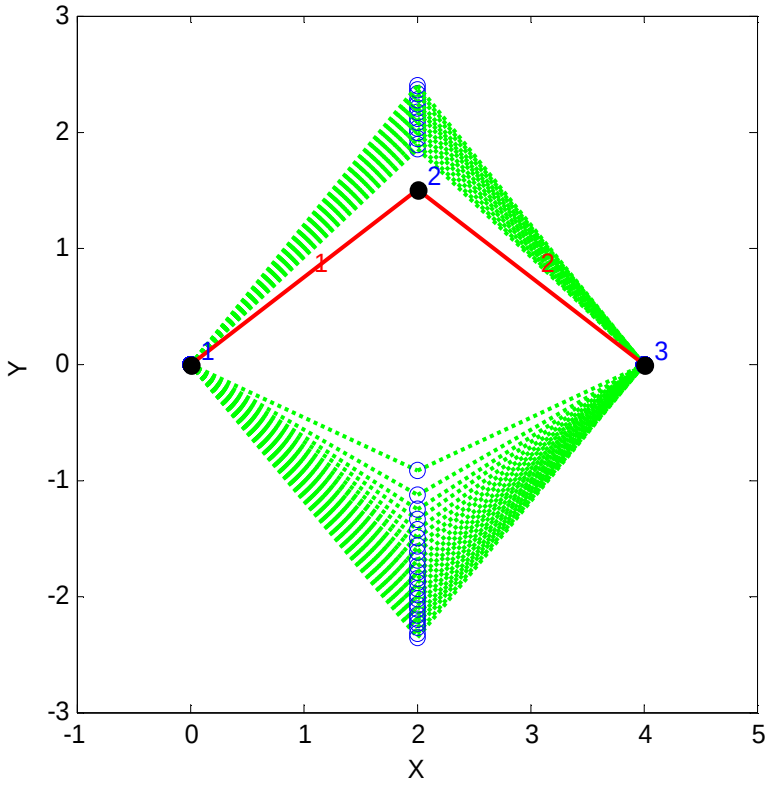
Figura 6.2. Trajetória de equilíbrio da barra



Como pode ser observado pela figura acima, os resultados numéricos são gerados da esquerda para a direita no gráfico acima, e os resultados obtidos pelos dois métodos são coincidentes na maior parte da trajetória de equilíbrio, porém, quando os valores numéricos se aproximam de um pico da curva, com valor da carga próximo a $1x10^4$kN, nota-se que os resultados numéricos dão um salto para outro trecho da curva, onde a carga continua crescendo. Esse efeito, chamado de snap through, ocorre porque a modelagem acontece através do controle de cargas. Por isso, como a carga é programada para continuar crescendo, ela dá um salto no gráfico para onde este crescimento é possível.

Na Figura 6.3, é mostrada a estrutura indeformada (cor vermelha) e deformada (cor verde) gerada pelo programa como parte do resultado. Os pontos representados pelos círculos azuis são as posições do Nó 2 quando o programa converge em um determinado passo de carga. Nesta figura, os deslocamentos estão na mesma escala que a estrutura.

Figura 6.3. Visualização da estrutura indeformada e deformada



O trecho central onde não é mostrada a estrutura deformada, é justamente o trecho onde ocorreu o efeito snap through, onde o programa não conseguiu convergir.

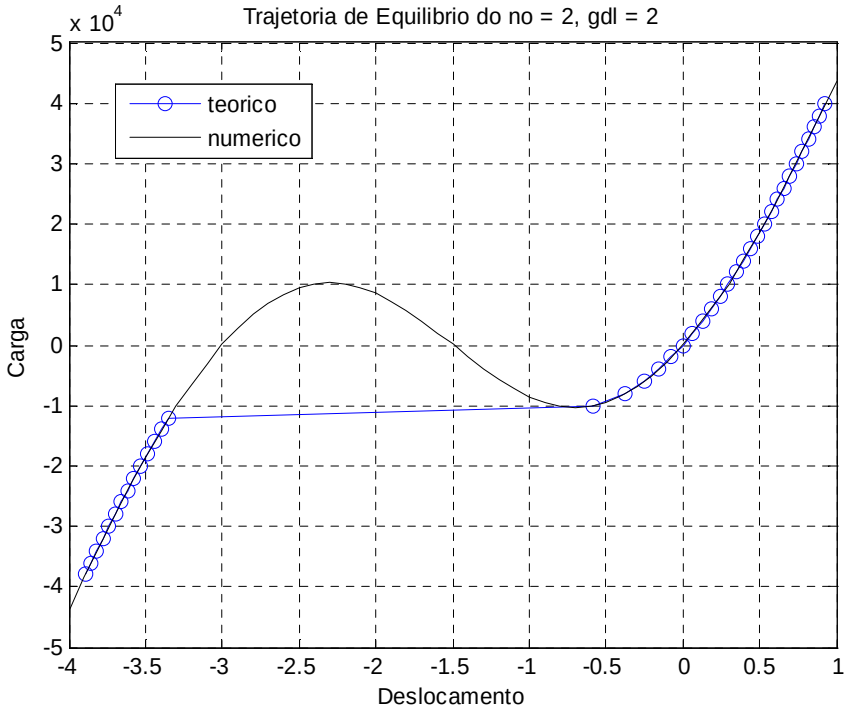
6.1.2. Teste de validação 1 – Parte 2

Na primeira parte deste teste de validação, o efeito de snap through apareceu durante a análise. Para investigar melhor este fenômeno numérico causado pelo método de Newton Raphson, realizou-se uma segunda etapa deste teste. Foi usada a mesma treliça do primeiro teste (Figura 6.1), também de material linear elástico, porém com o incremento do carregamento no sentido oposto, ou seja, o programa foi preparado para aplicar, primeiramente uma carga, no sentido para cima, aproximadamente de $4{,}5x10^{4}$kN, e variar o carregamento para baixo com o valor aproximado de $-4{,}5x10^{4}$kN. Sendo a = 2m, b = 1,5m, área da seção das barras é igual a $5x10^{-4}$ m^2 e Módulo de elasticidade igual a $2x10^{8}$MPa.

Para obter-se o resultado pelo programa, alterou-se o arquivo de entrada de dados com informações sobre material, dimensões dos elementos e seções, e carregamento aplicado, conforme mostrado no Anexo B.2.

A Figura 6.4 compara a trajetória de equilíbrio da estrutura calculada através dos resultados obtidos de forma analítica (linha de cor preta) com os dados numéricos (linha com círculos de cor azul).

Figura 6.4. Trajetória de equilíbrio da barra

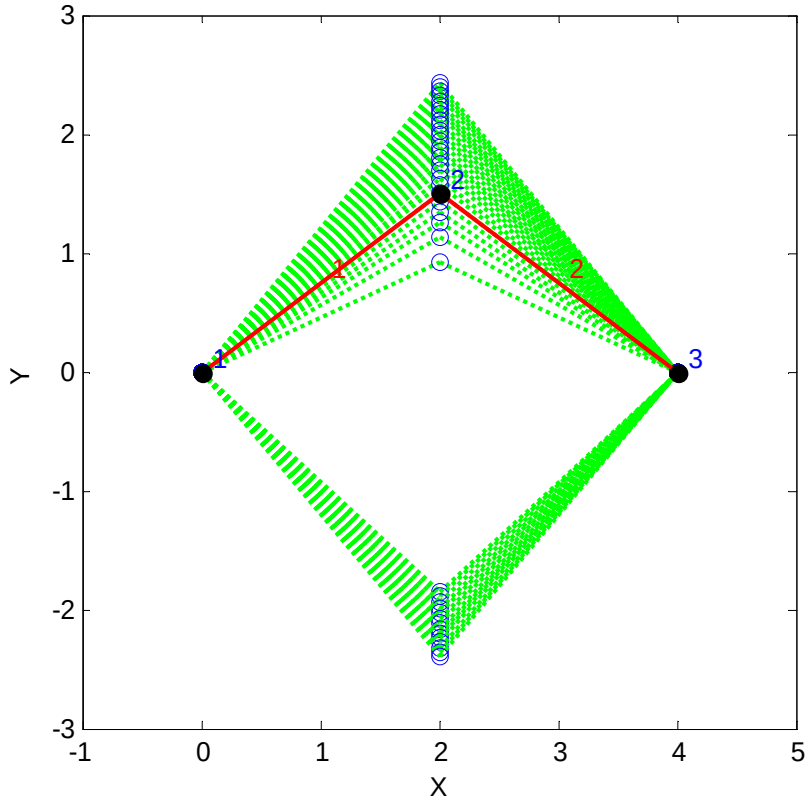


Como pode ser observado pela figura acima, os resultados numéricos são gerados da direita para a esquerda no gráfico acima, e os resultados obtidos pelos dois métodos são

coincidentes na maior parte da trajetória de equilíbrio, porém, no ponto onde a carga tem valor próximo a $-1x10^{4}$kN, nota-se, novamente, o efeito snap through. Nota-se neste gráfico que na etapa 2 deste teste de validação, alguns pontos de equilíbrio são diferentes dos encontrados na etapa 1.

Na Figura 6.5, é mostrada a estrutura indeformada e deformada gerada pelo programa com deformações na mesma escala que a estrutura. As posições da estrutura deformada coincidem com os novos pontos da trajetória de equilíbrio.

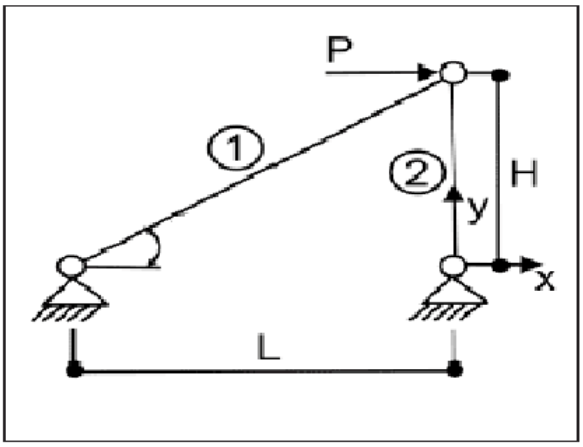
Figura 6.5. Visualização da estrutura indeformada e deformada



6.2. Teste de validação 2

No teste de validação dois é usada uma treliça com duas barras onde a barra 2 está na posição vertical, e carga concentrada no nó 2 na direção horizontal no sentido esquerda para direita, conforme Figura 6.6 , também de material linear elástico com módulo de elasticidade igual a $2x10^8$MPa em ambas as barras, onde L = 8 m e H = 10 m.

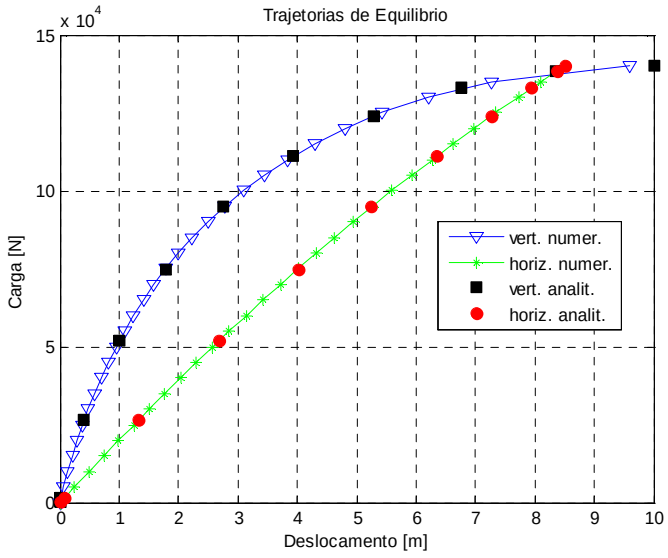
Figura 6.6. Visualização do esquema da estrutura



Para obter-se o resultado pelo programa, alterou-se o arquivo de entrada de dados com informações sobre material, dimensões dos elementos e seções, e carregamento aplicado, conforme mostrado no Anexo B.3.

A Figura 6.7 compara a trajetória de equilíbrio da estrutura, obtida através dos resultados calculado de forma analítica e numérica, do deslocamento do nó superior em ambos os eixos (horizontal e vertical), conforme legenda apresentada nesta figura.

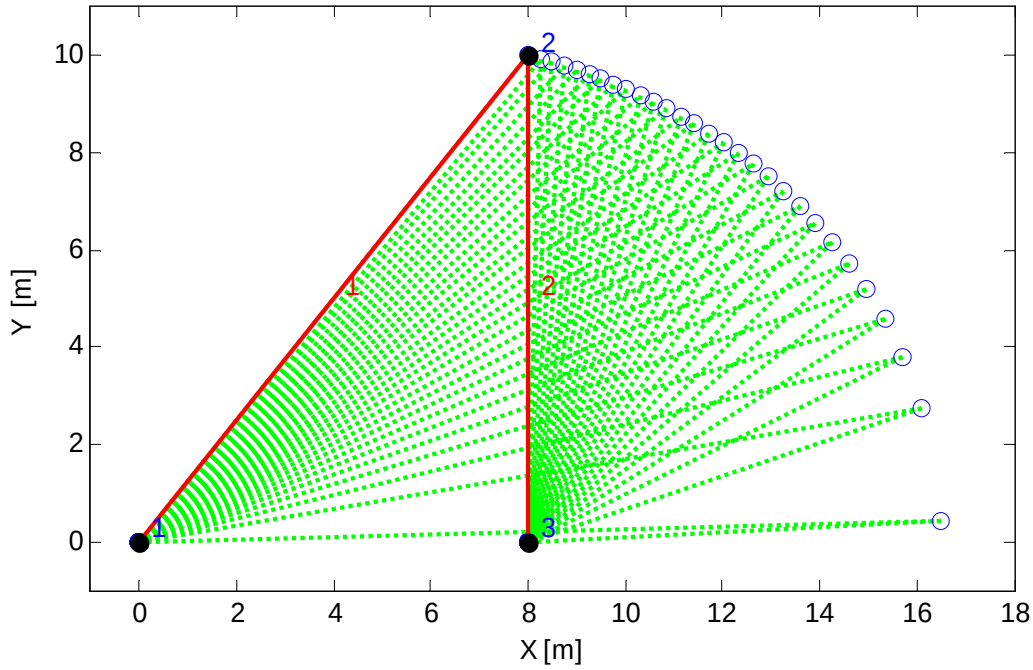
Figura 6.7. Trajetória de Equílibrio da Estrutura



Como pode ser observado pela figura acima, os resultados analíticos são dados apenas através de alguns pontos, e foram fornecidos por GRECO (2009). Os dados numéricos são mostrados por maior número de pontos que se ligando, formam uma curva. Observa-se, que esta curva coincide com os dados analíticos.

Na Figura 6.8, é mostrada a estrutura indeformada e deformada gerada pelo programa durante a análise. Vale ressaltar que as deformações apresentadas nesta figura estão na mesma escala que a estrutura.

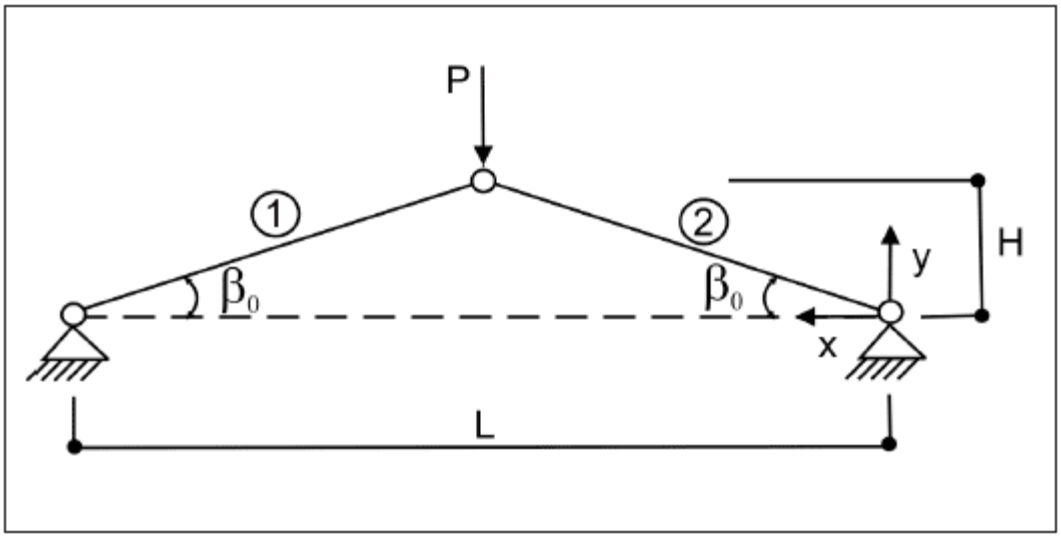
Figura 6.8. Estrutura indeformada e deformada



6.3. Teste de validação 3

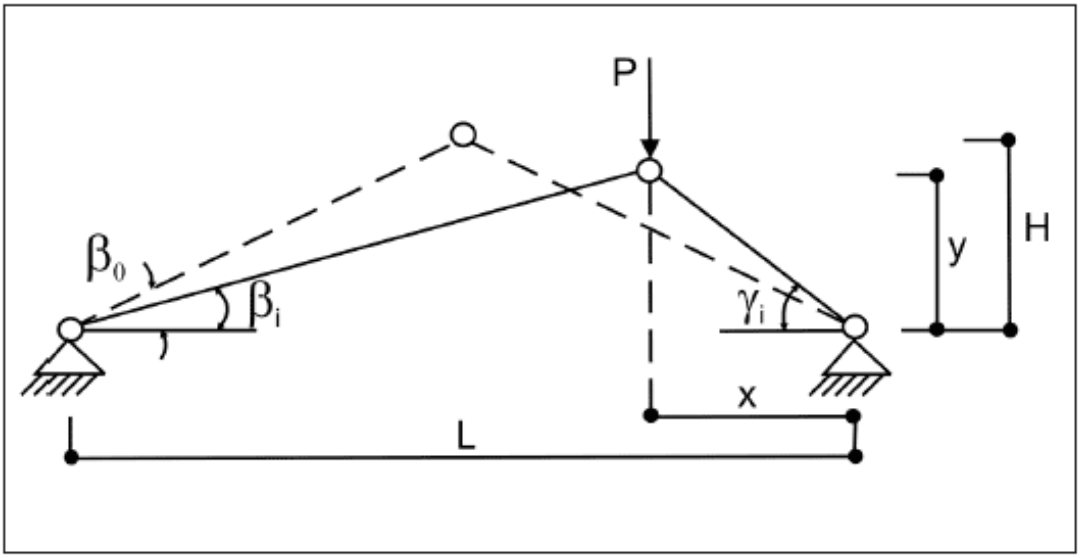
Este teste de validação consiste em analisar uma treliça de duas barras com geometria igual à treliça do primeiro teste de validação. A diferença está no material das barras, que neste caso, uma barra possui rigidez diferente da outra. Na primeira etapa do teste, as duas barras possuem material linear elástico, e na segunda etapa, uma das barras possui material não linear hiperelástico, e a outra material linear elástico. O esquema da treliça na posição indeformada e a posição da carga aplicada é dado na Figura 6.9.

Figura 6.9. Esquema da estrutura na posição indeformada



Esta estrutura, após a aplicação do carregamento, atinge o equilíbrio e toma a forma deformada conforme mostrado na Figura 6.10.

Figura 6.10. Esquema da estrutura na posição deformada



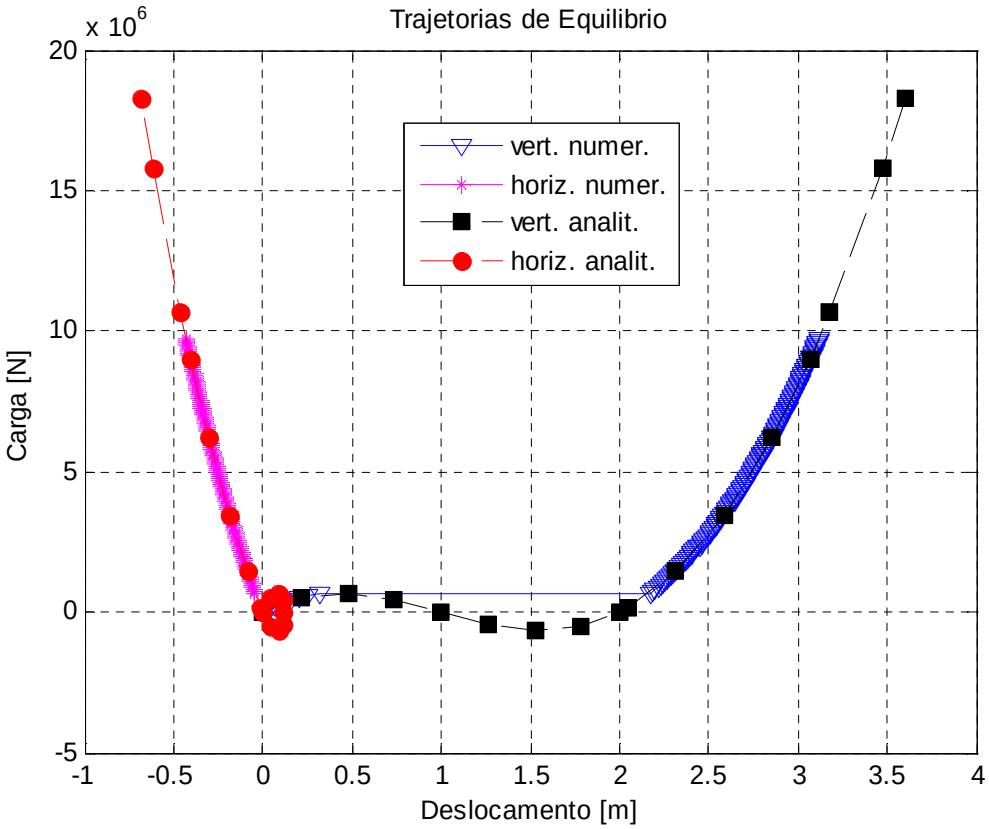
6.3.1. Teste de validação 3 – Parte 1

Nesta etapa testa-se o programa para a situação quando as barras possuem rigidez diferentes, apesar de ambas serem de material linear elástico. A barra 1 possui E_1A_1=80000kN, e a barra 2 tem E_2A_2=20000kN. Os valores do comprimento L é igual a 5m e altura H é igual a 1m.

Para obter-se o resultado pelo programa, alterou-se o arquivo de entrada de dados com informações sobre material, dimensões dos elementos e seções, e carregamento aplicado, conforme mostrado no Anexo B.4.

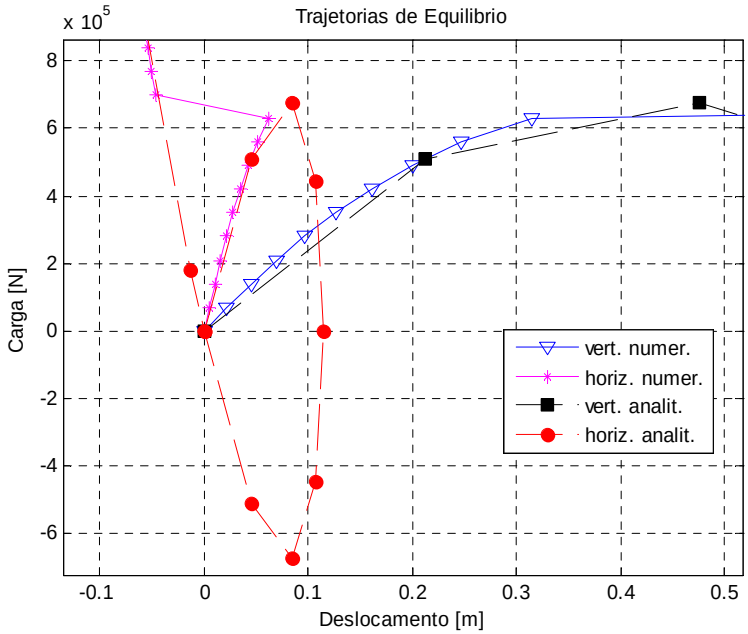
A Figura 6.11 compara a trajetória de equilíbrio da estrutura, obtida através dos resultados calculado de forma analítica e numérica, do deslocamento do nó superior em ambos os eixos (horizontal e vertical), conforme legenda apresentada nesta figura.

Figura 6.11. Trajetória de Equilíbrio da Estrutura



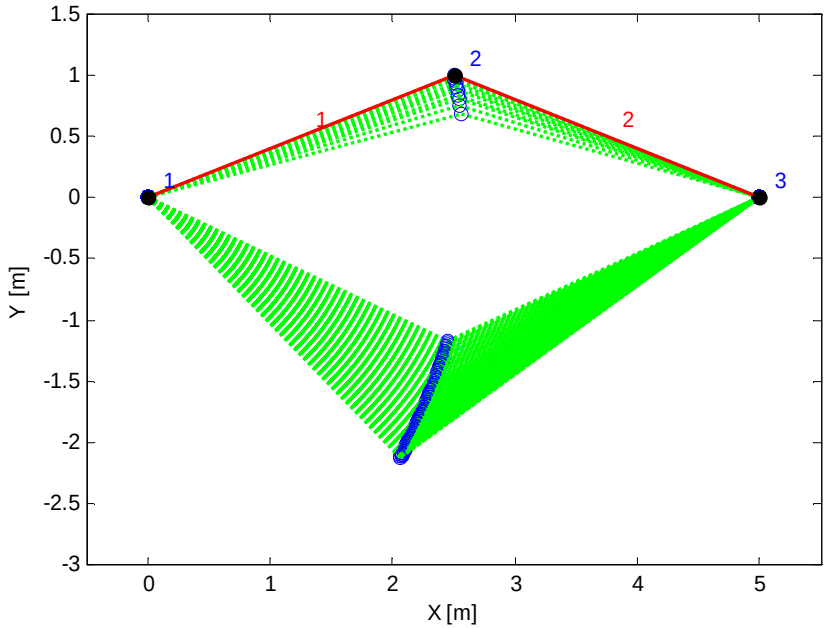
Ampliando-se a área inferior da Figura 6.11, obtem-se a Figura 6.12, que permite visualizar o comportamento da treliça no início do carregamento, detectando-se o efeito snap through.

Figura 6.12. Trajetória de Equilíbrio Ampliada



Na Figura 6.13, é mostrada a estrutura indeformada e deformada. Os pontos representados pelos círculos azuis são as posições do Nó 2 quando o sistema atinge um ponto de equilíbrio. Observa-se que o nó 2 , devido a menor rigidez da barra 2, desloca-se para direita quando as barras são comprimidas, e para esquerda, quando as barras são tracionadas.

Figura 6.13. Esquema da estrutura deformada e indeformada



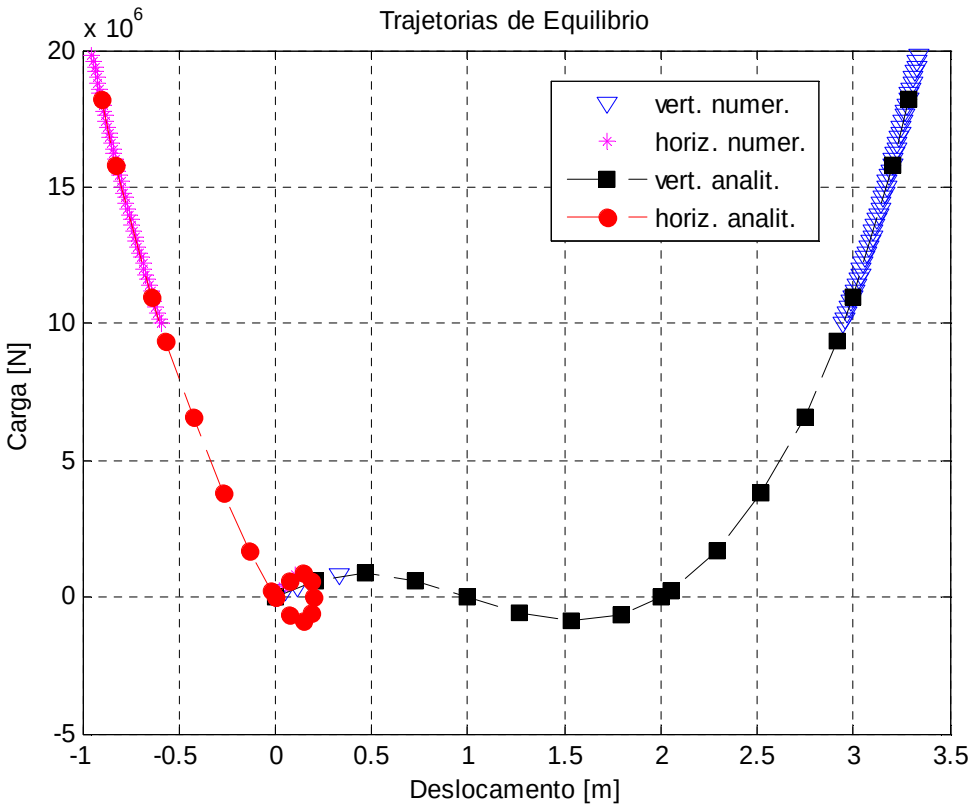
6.3.2. Teste de validação 3 – Parte 2

Nesta segunda parte deste teste de validação, muda-se apenas a equação que governa o comportamento da barra 1 que passar a ser não linear hiperelástico com lei constitutiva igual a $\sigma = E\sqrt{\varepsilon}$.

Para obter-se o resultado pelo programa, alterou-se o arquivo de entrada de dados com informações sobre material, dimensões dos elementos e seções, e carregamento aplicado, conforme mostrado no Anexo B.5.

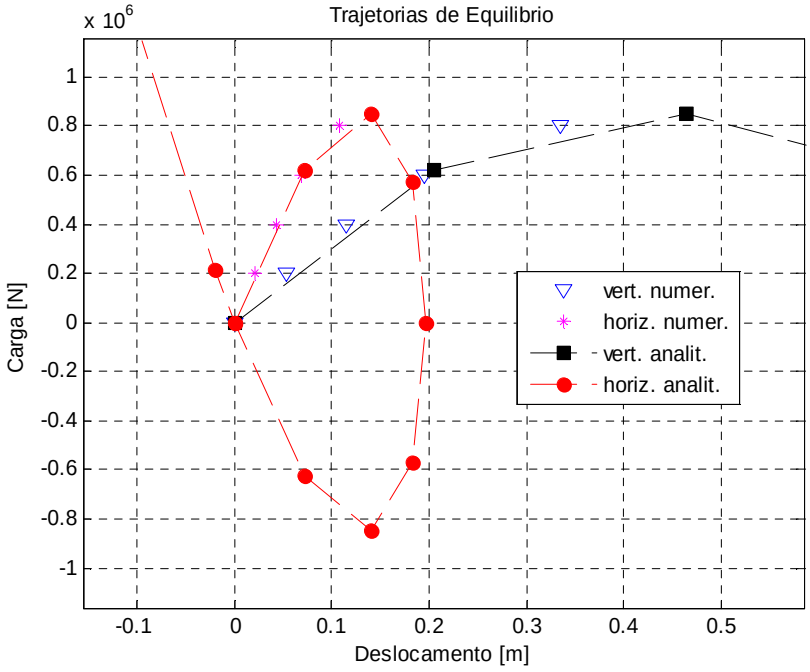
A Figura 6.14 compara a trajetória de equilíbrio da estrutura, obtida através dos resultados calculado de forma analítica e numérica, do deslocamento do nó superior em ambos os eixos (horizontal e vertical), conforme legenda apresentada nesta figura.

Figura 6.14. Trajetória de Equilíbrio da Estrutura



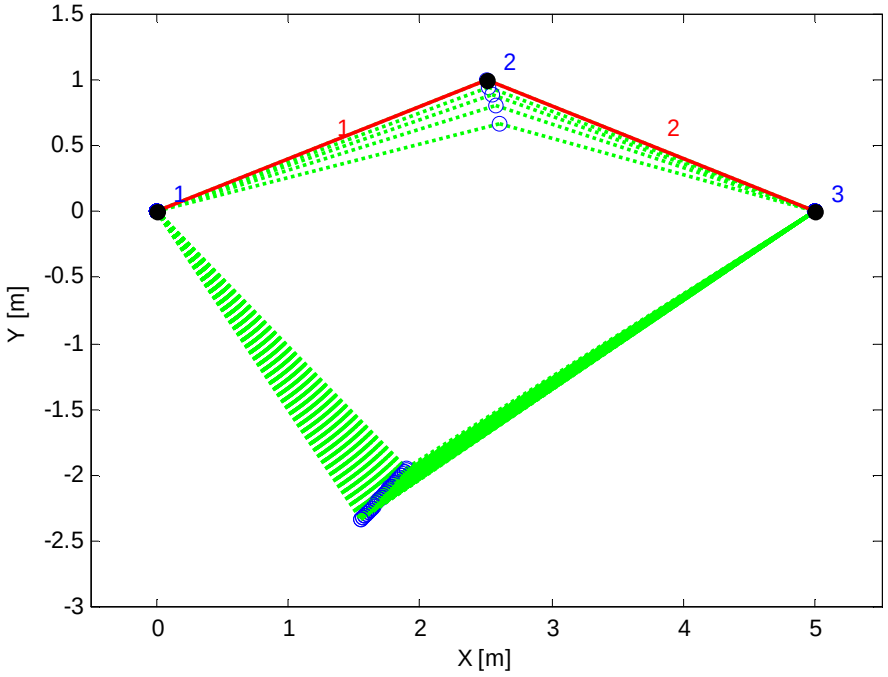
Ampliando-se a área inferior da Figura 6.14, obtem-se a Figura 6.15, que permite visualizar o comportamento da treliça no início da aplicação das cargas.

Figura 6.15. Trajetória de Equilíbrio Ampliada



Na Figura 6.16, é mostrada a estrutura indeformada e deformada da estrutura, indicando o deslocamento do nó 2 durante a análise.

Figura 6.16. Esquema da estrutura deformada e indeformada



6.4. Teste de validação 4

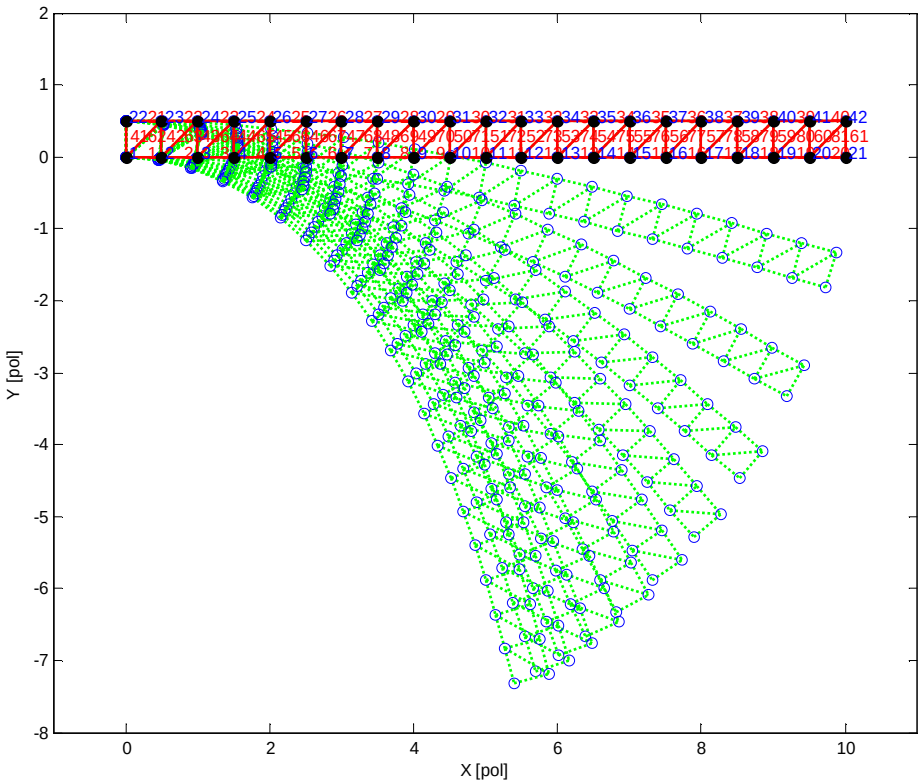
Este teste de validação compara os resultados obtidos através do programa com os resultados de YAW(2009) que também fez um programa para análise não linear utilizando a formulação co-rotacional.

A estrutura modelada trata-se de uma treliça com comprimento total de 10 polegadas, sendo formada por 42 nós e 81 barras, sendo que todas as barras possuem 0,5 polegadas de comprimento. O material é linear elástico.

Esta treliça possui dois apoios de segundo gênero em uma extremidade, e na outra possui uma carga concentrada.

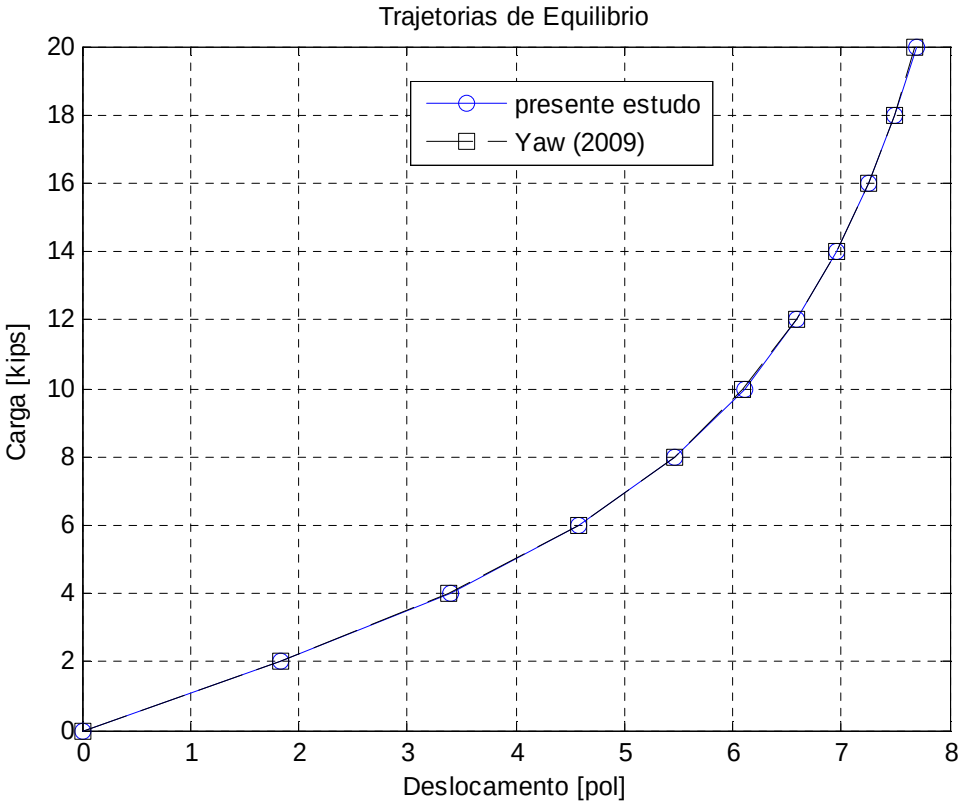
Na Figura 6.17, é mostrada a estrutura indeformada (cor vermelha) e deformada (cor verde) gerada pelo programa como parte do resultado. Os pontos representados pelos círculos azuis são as posições do Nó 2 quando o programa converge em um determinado passo de carga.

Figura 6.17. Esquema da estrutura deformada e indeformada



A Figura 6.18 compara os resultados obtidos pelos dois estudos. Observa-se que o nível de coincidência entre os resultados é muito alto.

Figura 6.18. Teste 6 - Trajetória de Equilíbrio



7. CONCLUSÃO

Este estudo tinha como objetivo o desenvolvimento de um sistema computacional com a funcionalidade de fazer análise estrutural de treliças planas considerando a não linearidade física e geométrica utilizando a formulação co-rotacional. Esta formulação permite a utilização de um sistema adicional, chamado básico, onde os deslocamentos ocorrem somente no eixo axial, e não possui deslocamentos de corpo rígido.

A linguagem de programação utilizada para o desenvolvimento foi MatLab, utilizando-se também, durante as etapas de análise e projeto do sistema, conceitos e filosofia de programação orientada a objeto, com a finalidade de identificar e organizar as funções de facilitar a compreensão do programa, e possibilitar que futuros incrementos com novas funcionalidades no programa sejam feitos com o mínimo de alteração no código existente.

Foram feitos seis testes de validação do programa, comparando os dados gerados por este sistema com dados encontrados em artigos científicos, que foram calculados de forma analítica e numérica.

Nos testes de validação foram utilizados casos com treliças considerando não linearidade física e geométrica, e casos considerando apenas a não linearidade geométrica.

A comparação mostrou que os resultados calculados por este sistema computacional e os resultados retirados da literatura coincidiram, confirmando que o programa desenvolvido é eficaz e com aplicabilidade para o objetivo proposto.

O método de encontrar raízes utilizado, Newton-Raphson, produziu bons resultados, exceto em trechos da trajetória de equilíbrio, onde não foi possível a convergência.

Como sugestão para trabalhos futuros indica-se:

- implementar um método de solução de equações que possam percorrer toda trajetória de equilíbrio, como o método do comprimento de arco e o minimal residual displacement method;
- Incrementar o programa para treliças espaciais;
- Adicionar a funcionalidade de ter como dado de saída um arquivo de texto com resultados principais.
- Adicionar novas leis constitutivas de materiais não lineares

REFERÊNCIAS BIBLIOGRÁFICAS

Aranha, Gandhy Yeddo da Rocha; DE SOUZA, Remo Magalhães. **Elemento Finito de Barra para Análise Geométrica Não-Linear Estática e Dinâmica através da Formulação Co-rotacional**. SIAM Review, v. 24, p. 427-440, 2004.

Aranha, Gandhy Yeddo da Rocha. **A Formulação de um Elemento Finito de Barra Para Análise Dinâmica Não-Linear Geométrica, com Aplicação a Cabos de Linhas Aéreas de Transmissão de Energia Elétrica.** 2003. Dissertação (Mestrado) - Universidade Federal do Pará, 2003.

BARON, Frank; VENKATESAN, Mahadeva S. **Nonlinear analysis of cable and truss structures**. Journal of the Structural Division, v. 97, n. 2, p. 679-710, 1971.

BOSCO, Melina et el. **Improvement of the Model Proposed by Menegotto and Pinto for Steel**. Second European Conference on Earthquake Engineering and Seismology, Istanbul, 2014

DE SOUZA, R. M. **Desenvolvimento de um Programa para Prognóstico, Monitoramento e Controle de Vibração em Condutores de LT's na Região Norte**. Projeto P&D, Convênio: UFPa/Eletronorte via ANEEL, 2002.

DEITEL, H. M.; DEITEL, P. J. **Java Como Programar**. 40 ed., Prentice Hall, 2002.

FAROUGHI, Shirko; KHODAPARAST, Hamed Haddad; FRISWELL, Michael I. **Non-linear dynamic analysis of tensegrity structures using a co-rotational method**. International Journal of Non-Linear Mechanics, v. 69, p. 55-65, 2015.

FILIPPOU F.C., POPOV E.P., BERTERO V.V. (1983) **Effects of bond deterioration on hysteretic behavior of reinforced concrete joints**, Report UCB/EERC 83/19

FUJII, Gilson. **Análise de Estruturas Tridimensionais: Desenvolvimento de uma Ferramenta Computacional Orientada para Objetos**. 1997. Tese de Doutorado. Universidade de São Paulo.

GIUFFRÉ A., PINTO P.E. (1970). **Il Comportamento del Cemento Armato per Sollecitazioni Cicliche di Forte Intensità**, Giornale del Genio Civile

GOLDBERG J.E., RICHARD R.M. (1963) **Analysis of nonlinear structures**. Journal of Structural Division (ASCE); 89(4): 333-351

GRECO, M., VICENTE, C. E. R. **Analytical solutions for geometrically nonlinear trusses.** Rev. Esc. Minas. vol.62 no.2. Ouro Preto. Apr.\June, 2009

JAGANNATHAN, Dharapuram S.; CHRISTIANO, Paul P.; EPSTEIN, Howard I. **Nonlinear analysis of reticulated space trusses**. Journal of the Structural Division, v. 101, n. 12, p. 2641-2658, 1975.

LE, Thanh Nam. **Corotational formulation for nonlinear analysis of flexible beam structures**. 2012

LEU, Liang-Jenq, and Yeong-Bin Yang. "**Effects of rigid body and stretching on nonlinear analysis of trusses.**" Journal of Structural Engineering 116.10 (1990): 2582-2598

MARQUES, Severino Pereira Cavalcanti. **Análise não linear física e geométrica de pórticos espaciais**. 1990.

MENEGOTTO M., PINTO P.E. (1972) **Method of analysis for cyclically loaded R.C. frames including changes in geometry and non-elastic behavior of elements under combined normal force and bending**, Istituto di Scienza e Tecnica delle Costruzioni, University of Rome, Report 32, October

MENEGOTTO M., PINTO P.E. (1973) **Method of anaysis for cyclically loaded reinforced concrete plane frames including changes in geometry and non-elastic behavior of elements under combined normal force and bending**, IABSE Symposium of Resistance and Ultimate Deformability of Structures Acted on by Well-Defined Repeated Loads, International Association of Bridge and Structural Engineering, Libson, Portugal, Vol. 13: 15-22.

PEREIRA, A. M. B. **Avanços na Visualização, Análise Não-Linear e Programação com o Método dos Elementos de Contorno.** 2004. Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo, 2004.

SAFFARI, H., M. J. Fadaee, and R. Tabatabaei. **"Nonlinear analysis of space trusses using modified normal flow algorithm."** Journal of structural engineering 134.6 (2008): 998-1005.

SMITH, Ben. **Object-Oriented Programming. In: Advanced ActionScript 3**. Apress, 2015. p. 1-23.

VINCENZI, Auri Marcelo Rizzo. **Orientação a objeto: definição, implementação e análise de recursos de teste e validação**. 2004. Tese de Doutorado. Universidade de São Paulo (USP). Instituto de Ciências Matemáticas e de Computação de São Carlos.

WIKIPEDIA, The Free Encyclopedia. Disponível em: <http://en.wikipedia.org/wiki/Main_Page>. Acesso contínuo, 2005.

YANG, Yeong-Bin; CHIOU, Hwa-Thong. **Rigid body motion test for nonlinear analysis with beam elements**. Journal of engineering mechanics, v. 113, n. 9, p. 1404-1419, 1987.

YAW, L. L., **2D Co-rotational Truss Formulation.** Walla Walla University. 2009

ZAMBONI, Lincoln C.; BARROS, Edson AR; PAMBOUKIAN, Sergio VD. **A Programação Orientada a Objetos como Ferramenta para o Aprendizado e Auxílio em Projetos de Engenharia**. In: COBENGE-Congresso Brasileiro de Ensino de Engenharia. Anais. 2005.

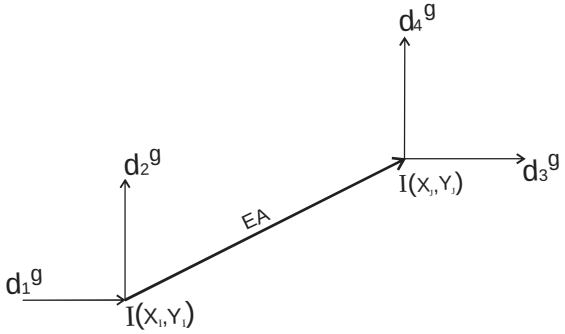
ANEXOS

ANEXO A

Passo a passo da formulação co-rotacional

Para determinar o vetor de forças resistente $\{\rho^g\}$ e a matriz tangente $[K^g]$ de uma barra de treliça sujeita a grandes deslocamentos, considerando material linear elástico e empregando a formulação co-rotacional

Dados:



Coordenadas:

$I = (X_I, Y_I)$

$J = (X_J, Y_J)$

Seção:

E = Módulo de Elasticidade

A = Área da seção transversal da barra

Deslocamento iniciais:

$$\{d^g\} = \begin{Bmatrix} d_1^g \\ d_2^g \\ d_3^g \\ d_4^g \end{Bmatrix}$$

Passo a passo para solução do problema:

1º passo: montagem do vetor para o ponto I

$$\overrightarrow{X_I} = \begin{Bmatrix} X_I \\ Y_J \end{Bmatrix}$$

2º passo: montagem do vetor para o ponto J

$$\overrightarrow{X_J} = \begin{Bmatrix} X_J \\ Y_J \end{Bmatrix}$$

3º passo: calcular o vetor $\overrightarrow{X_{IJ}}$

$$\overrightarrow{X_{IJ}} = \overrightarrow{X_J} - \overrightarrow{X_I}$$

4° passo: calcular o comprimento inicial da barra

$$\ell = \left[\overrightarrow{X_{IJ}}\right]$$

5° passo: calcular vetor unitário $\widehat{e_1}$

$$\widehat{e_1} = \frac{\overrightarrow{X_{IJ}}}{\left|\overrightarrow{X_{IJ}}\right|} = \frac{\overrightarrow{X_{IJ}}}{\ell}$$

6° passo: calcular vetor unitário $\widehat{e_2}$

$$\widehat{e_2} = \begin{Bmatrix} e_2x \\ e_2y \end{Bmatrix} = \begin{Bmatrix} -e_1y \\ e_1x \end{Bmatrix}$$

7° passo: calcular vetor $\overrightarrow{U_I}$

$$\overrightarrow{U_I} = \begin{Bmatrix} d_{1g} \\ d_{2g} \end{Bmatrix}$$

8° passo: calcular vetor $\overrightarrow{U_J}$

$$\overrightarrow{U_J} = \begin{Bmatrix} d_{3g} \\ d_{4g} \end{Bmatrix}$$

9° passo: calcular vetor $\overrightarrow{U_{IJ}}$

$$\overrightarrow{U_{IJ}} = \overrightarrow{U_J} - \overrightarrow{U_I}$$

10° passo: calcular vetor X_{IJ}^f

$$X_{IJ}^f = X_{IJ} + U_{IJ}$$

11° passo: calcular o comprimento final ℓ^f

$$\ell^f = \left|X_{IJ}^f\right| = \sqrt{\left(\vec{X}_{IJ}^f\right)\left(\vec{X}_{IJ}^f\right)}$$

12° passo: calcular o vetor unitário do sistema básico $\hat{e}_1^f$

$$\hat{e}_1^f = \frac{\vec{X}_{IJ}^f}{\left|\vec{X}_{IJ}^f\right|} = \frac{\vec{X}_{IJ}^f}{\ell^f}$$

13° passo: calcular o vetor unitário do sistema básico $\hat{e}_2^f$

$$\hat{e}_2^f = \begin{Bmatrix} e_{2x}^f \\ e_{2y}^f \end{Bmatrix} = \begin{Bmatrix} -e_{1y}^f \\ e_{1x}^f \end{Bmatrix}$$

14° passo: calcular o alongamento da barra $\Delta\ell$

$\Delta\ell = \ell^f - \ell$ (pode haver considerável arredamento)

ou

$\Delta\ell = \frac{\left(2\vec{X}_{IJ} + \vec{U}_{IJ}\right)^T . \vec{U}_{IJ}}{\ell_f + \ell}$ (mais preciso)

15° passo: determinar d^f

$$d^f = \Delta\ell$$

16° passo: Calcular K^b (material linear elástico)

$$K^b = E\,\frac{A}{\ell}$$

17° passo: Calcular ρ_1^b

$$\rho_1^b = k^b . d_1^b$$

18° passo: Determinar $[T]$

$$[T] = [-\cos \propto^f \; -\text{sen} \propto^f \; \cos \propto^f \; \text{sen} \propto^f]$$

$$[T] = \left[\left\{-\hat{e}_1^f\right\}^T \left\{\hat{e}_1^f\right\}^T\right]$$

19° passo: Calcular vetor de forças $\{\rho^g\}$

$$\{\rho^g\} = [T]_{1x4}^T \{\rho^b\}_{4x1}$$

20° passo: Calcular a matriz $[S]$

$$[S] = \langle \text{sen}\,\alpha^f \left| -\cos \alpha^f \right| - \sin \alpha^f \mid \cos \alpha^f \rangle$$

21° passo: Determinar a matriz $[G]$

$$[G] = [S]^T [S]$$

$$[G] = \begin{bmatrix} s^2 & -cs & -s^2 & cs \\ -cs & c^2 & cs & -c^2 \\ -s^2 & cs & s^2 & -cs \\ cs & c^2 & -cs & c^2 \end{bmatrix}$$

22° passo: Determinar a matriz de rigidez da barra em relação ao sistema global $[K^g]$

$$[K^g] = \frac{1}{\ell^f}[G]\rho_1^b + T^T[K^b] \;\; [T]$$

Anexo B

Neste anexo são apresentados os arquivos de entrada de dados utilizados nos seis testes de validação. Nestes arquivos são inseridos todos os dados de entrada necessários para a realização da análise não linear de treliças considerando o método co-rotacional.

Os códigos foram copiados diretamente do editor de texto do MatLab, com isso foram mantidas todas as formatações padrões deste editor.

ANEXO B.1 – Teste de validação 1 – parte 1

```
% Universidade Federal do Para - UFPa
% Centro Tecnologico - CT
% Programa de Pós-Graduação em Engenharia Civil
% Arquivo exemplo para entrada de dados - inpTrelica01.m

function inpTrelica01
close all;  % fecha todas as figuras
% dados de entrada
numNos = 3       % numero de nos da estrutura
numElems = 2     % numero de elementos da estrutura

a  = 2.0;
b  = 1.5;
L0 = sqrt(a^2+b^2);

% coordenadas dos nos
%          X   Y
coords = [0.0 0.0;     % no 1
          a   b;     % no 2
          2*a 0.0];     % no 3



% restricoes nodais (condicoes de apoio)
%         Cx Cy
restrs = [1  1; % no 1
          0  0;   % no 2
          1  1];  % no 3

% cargas nodais
%           Fx Fy
cargasNos = [0 0;   % no 1
             0 10000;   % no 2 %MN
             0 0];  % no 3

% deslocamentos nodais
%           Dx Dy
deslocNos = [0 0;   % no 1
             0 0;   % no 2
             0 0]; % no 3

% parametros da analise (armazenado em um estrutura)
param.numPassos = 38;   % numero de passos de carga
param.TOL = 1e-12;      % tolerancia para o algoritmo de newton-raphson
param.numItersMax = 15; % numero de iteraçoes maxima no algoritmo de
Newton-Raphson
param.noChave = 2;      % no chave para o tracado da trajetoria de
equilibrio
param.gdlChave = 2;     % grau de liberdade chave para o tracado da
trajetoria
                        % de equilibrio
param.lambda0 = -3.6;
param.delta_lambda = 0.2;



% conectividade dos elementos
%         noI noJ
conect = [1   2;    % elemento 1
```

```
          2   3];    % elemento 2

% Dados do material
Ei = 2e8;  % [MPa] modulo de elasticidade inicial
fy = 1.769e3;   % [MPa] tensao de escoamento
bm = 0.9999;        %  parametro referente a rigidez do trecho final da
curva  (endurecimento)
phi = 5;        %  parametro referente a curvatura do trecho de transicao
%epslon = 0.002; % deformacao de interesse

% area da secao
A = 5e-4; % [m2]

% agrupa dados do material em um único vetor
mate = [Ei fy bm phi];

%            area e material
dadosElem = [A mate;   % elemento 1
             A mate];   % elemento 2


% chama o arquivo para analise estrutural
[Dm,Rm,S,lm] = AnaliseTreli2dNaoLinearFisicaV02 (numNos, numElems, coords,
restrs, cargasNos, deslocNos, conect, dadosElem, param);

% imprime os resultados da analise
%ImprimeResultadosTrelica2dv3(numNos, numElems, coords, restrs, cargasNos,
deslocNos, conect, dadosElem,...
%                            'outTrelica01v2.m', Dm, Rm, S, param);
%
%desenha a estrutura deformada
escala = 1; % fator de escala para os deslocamentos
DesenhaTrelica2dDeformadav3(numNos, numElems, coords, conect, Dm, escala,
param);
% desenha a estrutura indeformada
%escala = 20;
DesenhaEstrut2d(numNos, numElems, coords, conect);
box on;

 axis([-1 5 -3 3]);

%PlotaTrajetoriaDeEquilibrio(param, cargasNos(2,2), Dm);
PlotaTrajetoriaDeEquilibriov2(param, cargasNos(2,2),lm, Dm);


i=1;
for d =-4:0.1:1
    delta(i)=d;
    Lf = sqrt(a^2+(b+d)^2);
    Forca(i)= 2*Ei*A*((Lf-L0)*(b+d))/(L0*Lf);
    i=i+1;
end

plot (delta, Forca, 'k-');
hold on;
legend ('teorico', 'numerico');
grid;
box on;
```

ANEXO B.2 – Teste de validação 1 – parte 2

```
% Universidade Federal do Para - UFPa
% Centro Tecnologico - CT
% Departamento de Construcao Civil - DCC
% Programa de Pós-Graduação em Engenharia Civil

function inpTrelica01
close all;  % fecha todas as figuras
% dados de entrada
numNos = 3       % numero de nos da estrutura
numElems = 2     % numero de elementos da estrutura

a  = 2.0;
b  = 1.5;
L0 = sqrt(a^2+b^2);

% coordenadas dos nos
%          X   Y
coords = [0.0 0.0;     % no 1
          a   b;     % no 2
          2*a 0.0];     % no 3



% restricoes nodais (condicoes de apoio)
%         Cx Cy
restrs = [1  1; % no 1
          0  0;   % no 2
          1  1];  % no 3

% cargas nodais
%           Fx Fy
cargasNos = [0 0;   % no 1
             0 10000;   % no 2 %MN
             0 0];  % no 3

% deslocamentos nodais
%           Dx Dy
deslocNos = [0 0;   % no 1
             0 0;   % no 2
             0 0]; % no 3

% parametros da analise (armazenado em um estrutura)
param.numPassos = 40;   % numero de passos de carga
param.TOL = 1e-12;      % tolerancia para o algoritmo de newton-raphson
param.numItersMax = 15; % numero de iteraçoes maxima no algoritmo de
Newton-Raphson
param.noChave = 2;      % no chave para o tracado da trajetoria de
equilibrio
param.gdlChave = 2;     % grau de liberdade chave para o tracado da
trajetoria
                        % de equilibrio
param.lambda0 = 4;
param.delta_lambda = -0.2;



% conectividade dos elementos
%         noI noJ
conect = [1   2;    % elemento 1
```

```
          2   3];    % elemento 2

% Dados do material
Ei = 2e8;  % [MPa] modulo de elasticidade inicial
fy = 1.769e3;   % [MPa] tensao de escoamento
bm = 0.9999;        %  parametro referente a rigidez do trecho final da
curva  (endurecimento)
phi = 5;        %  parametro referente a curvatura do trecho de transicao
%epslon = 0.002; % deformacao de interesse

% area da secao
A = 5e-4; % [m2]

% agrupa dados do material em um único vetor
mate = [Ei fy bm phi];

%            area e material
dadosElem = [A mate;   % elemento 1
             A mate];   % elemento 2


% chama o arquivo para analise estrutural
[Dm,Rm,S,lm] = AnaliseTreli2dNaoLinearFisicaV02 (numNos, numElems, coords,
restrs, cargasNos, deslocNos, conect, dadosElem, param);

% imprime os resultados da analise
%ImprimeResultadosTrelica2dv3(numNos, numElems, coords, restrs, cargasNos,
deslocNos, conect, dadosElem,...
%                            'outTrelica01v2.m', Dm, Rm, S, param);
%
%desenha a estrutura deformada
escala = 1; % fator de escala para os deslocamentos
DesenhaTrelica2dDeformadav3(numNos, numElems, coords, conect, Dm, escala,
param);
% desenha a estrutura indeformada
%escala = 20;
DesenhaEstrut2d(numNos, numElems, coords, conect);
box on;

 axis([-1 5 -3 3]);

%PlotaTrajetoriaDeEquilibrio(param, cargasNos(2,2)/2, Dm);
PlotaTrajetoriaDeEquilibriov2(param, cargasNos(2,2),lm, Dm);


i=1;
for d =-4:0.1:1
    delta(i)=d;
    Lf = sqrt(a^2+(b+d)^2);
    Forca(i)= 2*Ei*A*((Lf-L0)*(b+d))/(L0*Lf);
    i=i+1;
end

plot (delta, Forca, 'k-');
hold on;
legend ('teorico', 'numerico');
grid;
box on;
```

ANEXO B.3 – Teste de validação 2

```
% Universidade Federal do Para - UFPa
% Centro Tecnologico - CT
% Programa de Pós-Graduação em Engenharia Civil

function inpTrelica01
close all;  % fecha todas as figuras
% dados de entrada
numNos = 3       % numero de nos da estrutura
numElems = 2     % numero de elementos da estrutura

a  = 8;
b  = 10;
L0 = sqrt(a^2+b^2);

% coordenadas dos nos
%          X   Y
coords = [0.0 0.0;     % no 1
          a   b;     % no 2
          a 0.0];     % no 3


% restricoes nodais (condicoes de apoio)
%         Cx Cy
restrs = [1  1; % no 1
          0  0;   % no 2
          1  1];  % no 3

% cargas nodais
%           Fx Fy
cargasNos = [0 0;   % no 1
             100000 0;  % no 2 %MN
             0 0];  % no 3

% deslocamentos nodais
%           Dx Dy
deslocNos = [0 0;   % no 1
             0 0;   % no 2
             0 0]; % no 3

% parametros da analise (armazenado em um estrutura)
param.numPassos = 29;   % numero de passos de carga
param.TOL = 1e-12;      % tolerancia para o algoritmo de newton-raphson
param.numItersMax = 15; % numero de iteraçoes maxima no algoritmo de
Newton-Raphson
param.noChave = 2;      % no chave para o tracado da trajetoria de
equilibrio
param.gdlChave = 2;     % grau de liberdade chave para o tracado da
trajetoria
                        % de equilibrio
param.lambda0 = 0;
param.delta_lambda = 0.05;


% conectividade dos elementos
%         noI noJ
conect = [1   2;    % elemento 1
          2   3];    % elemento 2
```

```
% Dados do material
Ei = 2e8;  % [MPa] modulo de elasticidade inicial
fy = 1.769e3;   % [MPa] tensao de escoamento
bm = 0.9999;        %  parametro referente a rigidez do trecho final da
curva  (endurecimento)
phi = 5;        %  parametro referente a curvatura do trecho de transicao
%epslon = 0.002; % deformacao de interesse

% area da secao
A = 5e-3; % [m2]

% agrupa dados do material em um único vetor
mate = [Ei fy bm phi];

%            area e material
dadosElem = [A mate;   % elemento 1
             A mate];   % elemento 2


% chama o arquivo para analise estrutural
[Dm,Rm,S,lm] = AnaliseTreli2dNaoLinearFisicaV02 (numNos, numElems, coords,
restrs, cargasNos, deslocNos, conect, dadosElem, param);

% imprime os resultados da analise
%ImprimeResultadosTrelica2dv3(numNos, numElems, coords, restrs, cargasNos,
deslocNos, conect, dadosElem,...
%                            'outTrelica01v2.m', Dm, Rm, S, param);
%
%desenha a estrutura deformada
escala = 1; % fator de escala para os deslocamentos
DesenhaTrelica2dDeformadav3(numNos, numElems, coords, conect, Dm, escala,
param);
% desenha a estrutura indeformada
%escala = 20;
DesenhaEstrut2d(numNos, numElems, coords, conect);
box on;

axis([-1 18 -1 11]);

%PlotaTrajetoriaDeEquilibrio(param, cargasNos(2,2), Dm);
PlotaTrajetoriaDeEquilibriov2(param, cargasNos(2,1),lm, -Dm, 'bv-');
param.noChave = 2;      % no chave para o tracado da trajetoria de
equilibrio
param.gdlChave = 1;     % grau de liberdade chave para o tracado da
trajetoria
                        % de equilibrio
PlotaTrajetoriaDeEquilibriov2(param, cargasNos(2,1),lm, Dm, 'g*-');
% i=1;
% for d =-4:0.1:1
%     delta(i)=d;
%     Lf = sqrt(a^2+(b+d)^2);
%     Forca(i)= 2*Ei*A*((Lf-L0)*(b+d))/(L0*Lf);
%     i=i+1;
% end

DX = [0 0.08236812 1.31521187 2.68340359 4.01052214 5.24877902 6.34782172
7.25968574 7.94301344 8.36620175 8.50949359];
```

```
DY = [0 0.0215127 0.40869869 1.00149329 1.78315971 2.76215903 3.93383827
5.27962604 6.76819118 8.35790251 10];
P = [1.283e-6 1.697 26.356 51.962 74.953 94.785 111.134 123.853 132.91
138.325 140.126]*1000;

h=plot (DY, P, 'ks');
set(h(1),'MarkerEdgeColor','k','MarkerFaceColor','k')
hold on;
h=plot (DX, P, 'ro');
set(h(1),'MarkerEdgeColor','r','MarkerFaceColor','r')
hold on;
% legend ('teorico', 'numerico');
grid;
box on;
legend ('vert. numer.', 'horiz. numer.', 'vert. analit.', 'horiz. analit.')
```

ANEXO B.4 – Teste de validação 3

```
% Universidade Federal do Para - UFPa
% Centro Tecnologico - CT
% Programa de Pós-Graduação em Engenharia Civil

function inpTrelica04
close all;  % fecha todas as figuras
% dados de entrada
numNos = 3       % numero de nos da estrutura
numElems = 2     % numero de elementos da estrutura

a  = 2.5;
b  = 1;
L0 = sqrt(a^2+b^2);

% coordenadas dos nos
%          X   Y
coords = [0.0 0.0;     % no 1
          a   b;     % no 2
          2*a 0.0];     % no 3


% restricoes nodais (condicoes de apoio)
%         Cx Cy
restrs = [1  1; % no 1
          0  0;   % no 2
          1  1];  % no 3

% cargas nodais
%           Fx Fy
cargasNos = [0 0;   % no 1
             0 -200000; % no 2 %MN
             0 0];  % no 3

% deslocamentos nodais
%           Dx Dy
deslocNos = [0 0;   % no 1
             0 0;   % no 2
             0 0]; % no 3

% parametros da analise (armazenado em um estrutura)
param.numPassos = 140;   % numero de passos de carga
param.TOL = 1e-12;      % tolerancia para o algoritmo de newton-raphson
param.numItersMax = 15; % numero de iteraçoes maxima no algoritmo de
Newton-Raphson
param.noChave = 2;      % no chave para o tracado da trajetoria de
equilibrio
param.gdlChave = 2;     % grau de liberdade chave para o tracado da
trajetoria
                        % de equilibrio
param.lambda0 = 0;
param.delta_lambda = 0.35;


% conectividade dos elementos
%         noI noJ
conect = [1   2;    % elemento 1
          2   3];    % elemento 2
```

```
% Dados do material
Ei = 2e8;  % [MPa] modulo de elasticidade inicial
fy = 1.769e3;   % [MPa] tensao de escoamento
bm = 0.9999;        %  parametro referente a rigidez do trecho final da
curva  (endurecimento)
phi = 5;        %  parametro referente a curvatura do trecho de transicao
%epslon = 0.002; % deformacao de interesse

% area da secao
A1 = 0.4; % [m2]
A2 = 0.1;

% agrupa dados do material em um único vetor
mate = [Ei fy bm phi];

%            area e material
dadosElem = [A1 mate;   % elemento 1
             A2 mate];   % elemento 2


% chama o arquivo para analise estrutural
[Dm,Rm,S,lm] = AnaliseTreli2dNaoLinearFisicaV02 (numNos, numElems, coords,
restrs, cargasNos, deslocNos, conect, dadosElem, param);

% imprime os resultados da analise
%ImprimeResultadosTrelica2dv3(numNos, numElems, coords, restrs, cargasNos,
deslocNos, conect, dadosElem,...
%                            'outTrelica01v2.m', Dm, Rm, S, param);
%
%desenha a estrutura deformada
escala = 1; % fator de escala para os deslocamentos
DesenhaTrelica2dDeformadav3(numNos, numElems, coords, conect, Dm, escala,
param);
% desenha a estrutura indeformada
%escala = 20;
DesenhaEstrut2d(numNos, numElems, coords, conect);
box on;

axis([-0.5 5.5 -3 1.5]);

%PlotaTrajetoriaDeEquilibrio(param, cargasNos(2,2), Dm);
PlotaTrajetoriaDeEquilibriov2(param, -cargasNos(2,2),lm, -Dm, 'bv-');
param.noChave = 2;      % no chave para o tracado da trajetoria de
equilibrio
param.gdlChave = 1;     % grau de liberdade chave para o tracado da
trajetoria
                        % de equilibrio
PlotaTrajetoriaDeEquilibriov2(param, -cargasNos(2,2),lm, Dm, 'm*-');
% i=1;
% for d =-4:0.1:1
%     delta(i)=d;
%     Lf = sqrt(a^2+(b+d)^2);
%     Forca(i)= 2*Ei*A*((Lf-L0)*(b+d))/(L0*Lf);
%     i=i+1;
% end
```

```
DX = [0 0.04506495 0.08485474 0.10798164 0.11554944 0.10798164 0.08485474
0.04506495 4.896916e-8 -0.0127411 -0.08965458 -0.18611601 -0.30175339 -
0.40731321 -0.46414358 -0.61688269 -0.6817887]
DY = [0 0.21271915 0.476024 0.73832902 1 1.26167098 1.523976 1.78728085
1.9999998 2.0515962 2.31677771 2.58301321 2.85155757 3.07021455 3.18185526
3.47302961 3.59705573]
P = [6.689e-4 510.228 674.002 443.35 4.799e-7 -443.35 -674.002 -510.228 -
0.000669 180.628 1484.897 3465.803 6200.907 9022.621 10684.35 15766.845
18270.317]*1000

h=plot (DY, P, 'k--s');
set(h(1),'MarkerEdgeColor','k','MarkerFaceColor','k')
hold on;
h=plot (DX, P, 'r--o');
set(h(1),'MarkerEdgeColor','r','MarkerFaceColor','r')
hold on;
% legend ('teorico', 'numerico');
grid;
box on;
legend ('vert. numer.', 'horiz. numer.', 'vert. analit.', 'horiz. analit.')
```

ANEXO B.5 – Teste de validação 4

```
% Universidade Federal do Para - UFPa
% Centro Tecnologico - CT
% Programa de Pós-Graduação em Engenharia Civil

function inpTrelica05
close all;  % fecha todas as figuras
% dados de entrada
numNos = 3       % numero de nos da estrutura
numElems = 2     % numero de elementos da estrutura

a  = 2.5;
b  = 1;
L0 = sqrt(a^2+b^2);

% coordenadas dos nos
%          X   Y
coords = [0.0 0.0;     % no 1
          a   b;     % no 2
          2*a 0.0];     % no 3


% restricoes nodais (condicoes de apoio)
%         Cx Cy
restrs = [1  1; % no 1
          0  0;   % no 2
          1  1];  % no 3

% cargas nodais
%           Fx Fy
cargasNos = [0 0;   % no 1
             0 -2000000;    % no 2 %MN
             0 0];  % no 3

% deslocamentos nodais
%           Dx Dy
deslocNos = [0 0;   % no 1
             0 0;   % no 2
             0 0]; % no 3

% parametros da analise (armazenado em um estrutura)
param.numPassos = 100;   % numero de passos de carga
param.TOL = 1e-12;      % tolerancia para o algoritmo de newton-raphson
param.numItersMax = 40; % numero de iteraçoes maxima no algoritmo de
Newton-Raphson
param.noChave = 2;      % no chave para o tracado da trajetoria de
equilibrio
param.gdlChave = 2;     % grau de liberdade chave para o tracado da
trajetoria
                        % de equilibrio
param.lambda0 = 0;
%param.delta_lambda = 0.1;
param.delta_lambda = 0.1;


% conectividade dos elementos
%         noI noJ
conect = [1   2;    % elemento 1
```

```
          2   3];    % elemento 2

% Dados do material
Ei = 2e8;  % [MPa] modulo de elasticidade inicial
fy = 1.769e3;   % [MPa] tensao de escoamento
bm = 0.9999;        %  parametro referente a rigidez do trecho final da
curva  (endurecimento)
phi = 5;        %  parametro referente a curvatura do trecho de transicao
%epslon = 0.002; % deformacao de interesse

% area da secao
A1 = 0.4; % [m2]
A2 = 0.1;

% agrupa dados do material em um único vetor
codmat1 = 0; % material hiperelastico  sigma=E*srqt(epslon);
codmat2 = 1; % material Giufre-Menogoto-Pinto

mate1 = [Ei fy bm phi codmat1];
mate2 = [Ei fy bm phi codmat2];

%            area e material
dadosElem = [A1 mate1;   % elemento 1
             A2 mate2];   % elemento 2


% chama o arquivo para analise estrutural
[Dm,Rm,S,lm,checkEquil] = AnaliseTreli2dNaoLinearFisicaV02 (numNos,
numElems, coords, restrs, cargasNos, deslocNos, conect, dadosElem, param);



% imprime os resultados da analise
%ImprimeResultadosTrelica2dv3(numNos, numElems, coords, restrs, cargasNos,
deslocNos, conect, dadosElem,...
%                             'outTrelica01v2.m', Dm, Rm, S, param);
%
%desenha a estrutura deformada
escala = 1; % fator de escala para os deslocamentos
DesenhaTrelica2dDeformadav4(numNos, numElems, coords, conect, Dm, escala,
param, checkEquil);
% desenha a estrutura indeformada
%escala = 20;
DesenhaEstrut2d(numNos, numElems, coords, conect);
box on;

axis([-0.5 5.5 -3 1.5]);

%PlotaTrajetoriaDeEquilibrio(param, cargasNos(2,2), Dm);
PlotaTrajetoriaDeEquilibriov3(param, -cargasNos(2,2),lm, -Dm, 'bv',
checkEquil);
param.noChave = 2;      % no chave para o tracado da trajetoria de
equilibrio
param.gdlChave = 1;     % grau de liberdade chave para o tracado da
trajetoria
                        % de equilibrio
PlotaTrajetoriaDeEquilibriov3(param, -cargasNos(2,2),lm, Dm, 'm*',
checkEquil);
% i=1;
```

```
% for d =-4:0.1:1
%     delta(i)=d;
%     Lf = sqrt(a^2+(b+d)^2);
%     Forca(i)= 2*Ei*A*((Lf-L0)*(b+d))/(L0*Lf);
%     i=i+1;
% end

% DX = [0 0.04506495 0.08485474 0.10798164 0.11554944 0.10798164 0.08485474
0.04506495 4.896916e-8 -0.0127411 -0.08965458 -0.18611601 -0.30175339 -
0.40731321 -0.46414358 -0.61688269 -0.6817887]
% DY = [0 0.21271915 0.476024 0.73832902 1 1.26167098 1.523976 1.78728085
1.9999998 2.0515962 2.31677771 2.58301321 2.85155757 3.07021455 3.18185526
3.47302961 3.59705573]
% P = [6.689e-4 510.228 674.002 443.35 4.799e-7 -443.35 -674.002 -510.228 -
0.000669 180.628 1484.897 3465.803 6200.907 9022.621 10684.35 15766.845
18270.317]*1000

 DX = [0 0.07274114 0.14059144 0.18209917 0.19608892 0.18209917 0.14059144
0.07274114 0 -0.01993828 -0.13578096 -0.27291633 -0.42909638 -0.56585936 -
0.6375627 -0.82416783 -0.90076282]
 DY = [0 0.20415791 0.46472562 0.73089246 1 1.26910754 1.53527438
1.79584209 2 2.04855328 2.29157875 2.52362992 2.74429806 2.91337095
2.99599961 3.20080955 3.28428241]
 P = [-7.9e-4 621.486 850.184 573.469 6.265e-8 -573.469 -850.184 -621.486 -
7.9e-4 211.736 1677.188 3785.261 6574.977 9366.676 10982.435 15832.141
18182.005]*1000

h=plot (DY, P, 'k--s');
set(h(1),'MarkerEdgeColor','k','MarkerFaceColor','k')
hold on;
h=plot (DX, P, 'r--o');
set(h(1),'MarkerEdgeColor','r','MarkerFaceColor','r')
hold on;
% legend ('teorico', 'numerico');
grid;
box on;
legend ('vert. numer.', 'horiz. numer.', 'vert. analit.', 'horiz. analit.')
%plot (checkEquil',-lm*cargasNos(2,2)','g-d');
```

ANEXO B.6 – Teste de validação 5

```
% Universidade Federal do Para - UFPa
% Centro Tecnologico - CT
% Programa de Pós-Graduação em Engenhria Civil

function inpTrelica04
close all;  % fecha todas as figuras
% dados de entrada

L=10;
dL=0.5;
H=0.5;
n=20;

% coordenadas dos nos
%          X   Y
% coords = [0.0 0.0;     % no 1
%           a   b;     % no 2
%           2*a 0.0];     % no 3

for i=1:n+1
  x(i)=dL*(i-1);
  y(i)=0;
  x(i+n+1)=dL*(i-1);
  y(i+n+1)=H;
end
numNos=2*(n+1);

coords = [x' y'];

% restricoes nodais (condicoes de apoio)
%         Cx Cy
% restrs = [1  1;   % no 1
%           0  0;   % no 2
%           1  1];  % no 3

restrs = zeros (numNos,2);
restrs(1,:)= [1 1];
restrs(n+2,:) = [1 0];


% cargas nodais
%           Fx Fy
% cargasNos = [0 0; % no 1
%              0 -2000000;  % no 2 %MN
%              0 0];  % no 3

cargasNos = zeros(numNos,2);
noCarga=numNos;
%noCarga = n+1;
%cargasNos(numNos,:) = [0 20];
cargasNos(noCarga,:) = [0 -20];


% deslocamentos nodais
%           Dx Dy
% deslocNos = [0 0; % no 1
%              0 0; % no 2
```

```
%              0 0]; % no 3

deslocNos = zeros(numNos,2);

% parametros da analise (armazenado em um estrutura)
param.numPassos = 11;   % numero de passos de carga
param.TOL = 1e-12;      % tolerancia para o algoritmo de newton-raphson
param.numItersMax = 15; % numero de iteraçoes maxima no algoritmo de
Newton-Raphson
param.noChave = noCarga;      % no chave para o tracado da trajetoria de
equilibrio
param.gdlChave = 2;     % grau de liberdade chave para o tracado da
trajetoria
                        % de equilibrio
param.lambda0 = 0;
param.delta_lambda = 0.1;



% conectividade dos elementos
%         noI noJ
% conect = [1   2;  % elemento 1
%           2   3];    % elemento 2

for i=1: n;
    % banzo inferior
    conect(i,:)=[i (i+1)];

    % banzo superior
    conect(n+i,:)=[(n+i+1) (n+i+2)];

    % montantes
    conect(2*n+i,:)=[i (n+i+1)];

    % diagonais
    conect(3*n+1+i,:)=[i (n+i+2)];
end
conect(3*n+1,:)=[(n+1) 2*(n+1)];

numElems = 4*n+1;

    %DesenhaEstrut2dv2(numNos, numElems, coords, conect);
%box on;

% Dados do material
Ei = 29000;     % [ksi] modulo de elasticidade inicial
fy = 1.769e3;   % [MPa] tensao de escoamento
bm = 0.9999;    %  parametro referente a rigidez do trecho final da curva
(endurecimento)
phi = 5;        %  parametro referente a curvatura do trecho de transicao
%epslon = 0.002; % deformacao de interesse

% area da secao
A = 0.1; % [m2]

% agrupa dados do material em um único vetor
%codmat = 0; % material hiperelastico  sigma=E*srqt(epslon);
codmat = 1; % material Giufre-Menogoto-Pinto

mate = [Ei fy bm phi codmat];
```

```
%            area e material
for i =1: numElems
  dadosElem(i,:) = [A mate];
end

% chama o arquivo para analise estrutural
[Dm,Rm,S,lm] = AnaliseTreli2dNaoLinearFisicaV02 (numNos, numElems, coords,
restrs, cargasNos, deslocNos, conect, dadosElem, param);

% imprime os resultados da analise
%ImprimeResultadosTrelica2dv3(numNos, numElems, coords, restrs, cargasNos,
deslocNos, conect, dadosElem,...
%                            'outTrelica01v2.m', Dm, Rm, S, param);
%
%desenha a estrutura deformada
escala = 1; % fator de escala para os deslocamentos
DesenhaTrelica2dDeformadav4(numNos, numElems, coords, conect, Dm, escala,
param);
% desenha a estrutura indeformada
%escala = 20;
DesenhaEstrut2dv2(numNos, numElems, coords, conect);
box on;
axis([-1 11 -8 2])
%axis([-0.5 5.5 -3 1.5]);

%PlotaTrajetoriaDeEquilibrio(param, cargasNos(2,2), Dm);
PlotaTrajetoriaDeEquilibriov3(param, -cargasNos(noCarga,2),lm, -Dm, 'bo-');

% i=1;
% for d =-4:0.1:1
%     delta(i)=d;
%     Lf = sqrt(a^2+(b+d)^2);
%     Forca(i)= 2*Ei*A*((Lf-L0)*(b+d))/(L0*Lf);
 %     i=i+1;
 % end

 % DX = [0 0.04506495 0.08485474 0.10798164 0.11554944 0.10798164
0.08485474 0.04506495 4.896916e-8 -0.0127411 -0.08965458 -0.18611601 -
0.30175339 -0.40731321 -0.46414358 -0.61688269 -0.6817887]
 % DY = [0 0.21271915 0.476024 0.73832902 1 1.26167098 1.523976 1.78728085
1.9999998 2.0515962 2.31677771 2.58301321 2.85155757 3.07021455 3.18185526
3.47302961 3.59705573]
 % P = [6.689e-4 510.228 674.002 443.35 4.799e-7 -443.35 -674.002 -510.228
-0.000669 180.628 1484.897 3465.803 6200.907 9022.621 10684.35 15766.845
18270.317]*1000

 %  DX = [0 0.07274114 0.14059144 0.18209917 0.19608892 0.18209917
0.14059144 0.07274114 0 -0.01993828 -0.13578096 -0.27291633 -0.42909638 -
0.56585936 -0.6375627 -0.82416783 -0.90076282]
 %  DY = [0 0.20415791 0.46472562 0.73089246 1 1.26910754 1.53527438
1.79584209 2 2.04855328 2.29157875 2.52362992 2.74429806 2.91337095
2.99599961 3.20080955 3.28428241]
 %  P = [-7.9e-4 621.486 850.184 573.469 6.265e-8 -573.469 -850.184 -
621.486 -7.9e-4 211.736 1677.188 3785.261 6574.977 9366.676 10982.435
15832.141 18182.005]*1000

 DY = [0 1.825 3.3889 4.5818 5.4569 6.0956 6.5826 6.9521 7.2495 7.4916
7.6815];
 P = [0 2 4 6 8 10 12 14 16 18 20];
```

```
h=plot (DY, P, 'k--s');
%set(h(1),'MarkerEdgeColor','k','MarkerFaceColor','k')
hold on;

legend ('presente estudo', 'Yaw (2009)');
grid;
box on;
```

ANEXO C

Neste anexo está disponibilizado todo código fonte de todas as funções do sistema computacional desenvolvido que realiza análise considerando não linearidade geométrica e física de treliças planas.

As códigos foram disponibilizados com a formatação original do MatLab para facilitar a identificação e compreensão, caso venham ser feitas futuras atualizações no programa

ANEXO C.1 – função `AnaliseTrelicadNaoLinear`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas
%
%function [Dm,Pm,F] = AnaliseTrelicadNaoLinear (numNos, numElems, coords,
restrs, cargasNos, deslocNos,...
%                                               conect, dadosElem, param);
%
% Dados de entrada:
%   numNos    = numero de nos
%   numElems  = numero de elementos
%   coords    = coordenadas dos nos
%   restrs    = restricoes nodais
%   cargasNos = cargas nodais
%   deslocNos = deslocamentos nodais
%   conect    = conectividade dos elementos (incidencia nodal)
%   dadosElem = dados do elemento (e.g., E, A)
%   param     = estrutura de dados contendo parametros da analise
%
% Parametros de saida:
%   Dm         = vetor de deslocamentos nodais
%   Pm         = vetor de forcas nodais (incluindo reacoes de apoio)
%   F         = matriz contendo as forcas locais dos elementos (esforcos
normal N)


% Parametros da estrutura
% nGrLib = numero de graus de liberdade da estrutura

function [Dm,Pm,F] = AnaliseTrelicadNaoLinear (numNos, numElems, coords,
restrs, cargasNos, deslocNos,...
                                    conect, dadosElem, param)

  % especifica parametros da estrutura
  NGLN  = 2                 % numero de graus de liberdade por no
  NGLE  = 4                 % numero de graus de liberdade por elemento
  NGLEst = NGLN * numNos  % numero total de graus de liberdade da estrutura

  % obtem os graus de liberdade da estrutura
  [NGLLE, gdl] = obtemGrausLibEstrut(numNos, restrs, NGLN)
```

```
  % monta vetor de carga nodal diretamente aplicado nos nós
  % e o vetor de deslocamentos nodais (considerando os deslocamentos
prescritos)
  Pext = zeros(NGLEst,1)
  D = zeros(NGLEst,1)
  D
  for n = 1: numNos
      n
      for m = 1: NGLN
          m
          g = gdl(n,m)
        Pext(g,1) = cargasNos(n,m)
        D(g,1) = deslocNos(n,m)
     end
  end


  % resolve o sistema iterativamente através do método de Newton-Raphson

  % atribui o vetor de deslocamentos como chute inicial para os graus de
liberdade
  % livres, e respeitando os
  % deslocamentos impostos nos nós

  DL = D(1:NGLLE)
  DR = D(NGLLE+1:NGLEst)

  % aplica o carregamento em passos de carga

  lambda = 0

  DiL = DL  % chute inicial para os deslocamentos livres

  for j = 1: param.numPassos
      j
    lambda = (j-1)/(param.numPassos-1)
    lambda
    DiR = lambda*DR; % impoe uma parcela lambda dos deslocamentos
prescritos nos nós
    DiR
    Pextj = lambda*Pext; % aplica uma parcela lambda da carga externa
    Pextj
    cont = 0  % contador do numero de iterações
    erro = 10 % define um valor grande para o erro, forçando pelo menos uma
iteração


    DeltaDi = zeros(NGLEst, 1) % não permite alteração na parte restringida
dos
                               % deslocamentos

    while (erro > param.TOL)

      % monta a matriz de rigidez global da estrutura
      Di = [DiL;
            DiR]

      [Pinti, Ki, N] = obtemVetForcasEMatRigEstrutura(NGLEst, numElems,
coords, conect, dadosElem, gdl, Di)
```

```
      % particiona e resolve o sistema de equacoes
      DeltaPi = Pextj - Pinti  % será considerada apenas a parte livre de
DeltaPi

      [DeltaPi, DeltaDi] = resolveSistEquacoes(NGLLE, NGLEst, Ki, DeltaPi,
DeltaDi)

      Di = Di + DeltaDi

      DiL = Di(1:NGLLE)

      erro = norm(DeltaDi)

      % outra possibilidade:  erro = norm(DeltaPi)
      % outra possibilidade:  erro = abs(DeltaDi'*DeltaPi); % norma de
energia

      cont = cont + 1


      if (cont > param.numItersMax)
          fprintf ('numero de iterações %d superou o valor máximo', cont);
          break;
      end

    end

    % obtem os esforços locais dos elementos para cada passo de carga
    %F(:,j) = obtemForcasSistemaBasicoElementos(NGLE, numNos, numElems,
coords, conect, dadosElem, gdl, Di)
    for el = 1: numElems
    el
    F(el) = N

    end

    % armazena as componentes do vetor P e D em matrizes (de forma analoga
as
    % matrizes forcasNos e deslocNos

    for no = 1 : numNos
        no
        for m = 1 : NGLN
            m
          g = gdl(no,m)
          Pm(no,m,j) = Pinti(g,:)
          Dm(no,m,j) = Di(g,:)
       end
    end
  end
return

%--------------------------------------------------------------------------
```

ANEXO C.2 – função `calcTensaoGMP`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% funcao para calculo da tensão e modulo de elasticidade tangente
% para o modelo de Giufre-Menegoto-Pinto, referente a um determinado
% valor de deformação

function [sigma, Et] = calcTensaoGMP(mate, eps)

% dados de entrada
% mate = vetor contendo os dados do material
% Ei = modulo de elasticidade inicial
% fy = tensao de escoamento
% b = parametro referente a rigidez do trecho final da curva
% (endurecimento)
% phi = parametro referente a curvatura do trecho de transicao
% epslon = deformacao de interesse

% dados de saida
% sigma = tensão no material referente a deformacao epslon
% Et = modulo de elasticidade tangente referente a deformacao epslon

Ei = mate(1)
fy = mate(2)
b = mate(3)
phi = mate(4)

epslon = abs(eps);

ey = fy/Ei  % deformação de escoamento
a=epslon/ey % fator entre a deformação e a deformacao de escoamento
R = 1/phi   % raio de curvatura do trecho de transição

sigma = fy * (b*a+(1-b)*a/((1+(a^phi))^R))
Et = fy *((b/ey)-(b-1)/(ey*(a^phi+1)^R) + epslon*(a^(phi-1))*(b-
1)/((ey^2)*(a^phi+1)^(R+1)))

sigma = sigma * sign(eps);
end
```

ANEXO C.3 – função `compOrientElem`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Faculdade de Engenharia Civil - FEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% calcula o comprimento e os cossenos diretores da barra

function [L, cost, sint] = compOrientElem(coordNoI, coordNoJ)

  % Dados de entrada
  % coordNoI = coordenadas do no I do elemento
  % coordNoJ = coordenadas do no J do elemento

  % Dados de saida
  dX = coordNoJ(1) - coordNoI(1)
  dY = coordNoJ(2) - coordNoI(2)

  L = sqrt(dX^2 + dY^2)

  cost = dX/L
  sint = dY/L

return

%-------------------------------------------------------------------------
-
```

ANEXO C.4 – função `DesenhaEstrut2d`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% function DesenhaEstrut2d(numNos, numElems, coords, conect);
% Parametros de entrada:

% numNos    = numero de nos
% numElems  = numero de elementos
% coords    = coordenadas dos nos
% restrs    = restricoes nodais
% cargasNos = cargas nodais
% conect    = conectividade dos elementos (incidencia nodal)

function DesenhaEstrut2d(numNos, numElems, coords, conect);

%cla;
%axis ('off');
%view([0 75]);

set(gcf,'DefaultLineColor','red')  % define a cor "default" das linhas
axis equal                         % mesma escala em x e y


% determina dimensoes minimas e maximas da estrutura
xmin = min(coords(:,1));
ymin = min(coords(:,2));
xmax = max(coords(:,1));
ymax = max(coords(:,2));

dx = xmax - xmin;
dy = ymax - ymin;

lado = max([dx,dy]);
delta = lado/10;

axis ([xmin-delta,xmax+delta,ymin-delta,ymax+delta]);

delta = lado/40;

for el = 1: numElems

   % determina os nos do elemento
   noI = conect(el,1);
   noJ = conect(el,2);

   % determina as coordenadas dos nos do elemento
   coordsElem(1,:) = coords(noI,:);
   coordsElem(2,:) = coords(noJ,:);

   coordMeio = (coordsElem(1,:) + coordsElem(2,:))/2;

   xLinha = coordsElem(1:2,1);
   yLinha = coordsElem(1:2,2);
```

```
   % desenha o elemento na configuracao indeformada

   H=line(xLinha,yLinha);
   set(H,'color',[1,0,0]);
   set(H,'LineWidth',[2]);

   elemNum = sprintf('%d',el);
   Ht=text(coordMeio(1)+delta,coordMeio(2)+delta,elemNum);
   set(Ht,'color',[1,0,0]);

end

hold on
for no=1: numNos
  xNo = coords(no,1);
  yNo = coords(no,2);

  plot(xNo,yNo,'k.','Markersize',18);
  noNum = sprintf('%d',no);
  Ht=text(xNo+delta,yNo+delta,noNum);
  set(Ht,'color',[0,0,1]);
end

xlabel ('X');
ylabel ('Y');

set(gcf,'DefaultLineColor','red')
```

ANEXO C.5 – função `DesenhaTrelica2dDeformada`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% function DesenhaTrelica2dDeformada(numNos, numElems, coords, conect, D,
escala);
% Parametros de entrada:

% numNos    = numero de nos
% numElems  = numero de elementos
% coords    = coordenadas dos nos
% restrs    = restricoes nodais
% cargasNos = cargas nodais
% conect    = conectividade dos elementos (incidencia nodal)
% Dm        = matriz contendo os deslocamentos nodais
% escala    = fator de escala para os deslocamentos

function DesenhaTrelica2dDeformada(numNos, numElems, coords, conect, Dm,
escala, param);

%cla;
%axis ('off');
%view([0 75]);

set(gcf,'DefaultLineColor','red')  % define a cor "default" das linhas
axis equal                         % mesma escala em x e y

for j = 1: param.numPassos
    % calcula as coordenadas deformadas
    for i = 1: numNos
       coordsDef(i,1) = coords(i,1) + Dm(i,1,j)*escala;
       coordsDef(i,2) = coords(i,2) + Dm(i,2,j)*escala;
    end

    % determina dimensoes minimas e maximas da estrutura
    xmin = min([coords(:,1)
                coordsDef(:,1)]);
    ymin = min([coords(:,2)
                coordsDef(:,2)]);
    xmax = max([coords(:,1)
                coordsDef(:,1)]);
    ymax = max([coords(:,2)
                coordsDef(:,2)]);

    dx = xmax - xmin;
    dy = ymax - ymin;

    lado = max([dx,dy]);
    delta = lado/10;

    axis ([xmin-delta,xmax+delta,ymin-delta,ymax+delta]);

    delta = lado/40;

    for el = 1: numElems
```

```
        % determina os nos do elemento
        noI = conect(el,1);
        noJ = conect(el,2);

        % determina as coordenadas dos nos do elemento
        coordsElem(1,:) = coordsDef(noI,:);
        coordsElem(2,:) = coordsDef(noJ,:);

        xLinha = coordsElem(1:2,1);
        yLinha = coordsElem(1:2,2);

        % desenha o elemento na configuracao indeformada

        H=line(xLinha,yLinha);
        set(H,'color',[0,1,0]);
        set(H,'LineWidth',[2]);
        set(H,'LineStyle', ':');

    end

    hold on
    for no=1: numNos
      xNo = coordsDef(no,1);
      yNo = coordsDef(no,2);

      plot(xNo,yNo,'bo');
    end
end
xlabel ('X');
ylabel ('Y');

set(gcf,'DefaultLineColor','red')
```

ANEXO C.6 – função `obtemForcasSistemaBasicoElementos`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% obtem os esforços locais dos elementos

%function F = obtemForcasSistemaBasicoElementos(NGLE, numNos, numElems,
coords, conect, dadosElem, gdl, D)


  for el = 1: numElems
    el
    % determina os nos do elemento
    noI = conect(el,1)
    noJ = conect(el,2)

    % determina as coordenadas dos nos do elemento
    coordNoI = coords(noI,:)  % copia a linha 'noI' da matriz coords na
matriz coordNoI
    coordNoJ = coords(noJ,:)  % copia a linha 'noJ' da matriz coords na
matriz coordNoJ

    % determina os graus de liberdade do elemento
    grLibElem = obtemGrausLibElemento(noI, noJ, gdl)

    % extrai os deslocamentos das barras, do vetor de deslocamentos da
estrutura
    for i = 1: 4
        dg(i,1) = D(grLibElem(i))
    end

    % obtem dados da seção e do material
    A = dadosElem(el,1)
    mate = dadosElem(el,2:5)'

    % calcula a matriz de rigidez do elemento em coordenadas globais
    [pb, kb, N]=obtemVetForcasEMatRigBasico(coordNoI, coordNoJ, A, mate,
db)

    % armazena as forcas locais na matris s para saida de resultados
    F(el) = N

    el
    N

  end
return
```

ANEXO C.7 – função `obtemGrausLibElemento`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Faculdade de Engenharia Civil - FEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% determina os graus de liberdade do elemento

function grLibElem = obtemGrausLibElemento(noI, noJ, gdl)

   grLibElem(1:2) = gdl(noI,:)
   grLibElem(3:4) = gdl(noJ,:)

return

%-------------------------------------------------------------------------
-
```

ANEXO C.8 – função `obtemGrausLibEstrut`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Faculdade de Engenharia Civil - FEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% obtem os graus de liberdade da estrutura

function [NGLLE, gdl, N] = obtemGrausLibEstrut(numNos, restrs, NGLN)

% NGLLE = numero de graus de liberdade livre da estrutura
% gdl   = graus de liberdade da estrutura

   gdl = zeros(numNos, NGLN)

   con = 0
   for no = 1: numNos
      for i = 1: NGLN
          if (restrs(no,i) == 0)
             con = con + 1
             gdl(no,i) = con
          end
      end
   end
   NGLLE = con
   for no = 1: numNos
      for i = 1: NGLN
         if (restrs(no,i) == 1)
             con = con + 1
             gdl(no,i) = con
         end
      end
   end
return

%-------------------------------------------------------------------------
-
```

ANEXO C.9 – função `obtemMatRotacao`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Faculdade de Engenharia Civil - FEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% calcula a matriz de rotacao do elemento (do sistema global para local)

function  [T, G] = obtemMatRotacao(cost, sint)

  c = cost
  s = sint

  T = [-c -s c s]

  S = [s -c -s c]

  G = S'*S


return
```

ANEXO C.10 – função `obtemMatRotacao`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Faculdade de Engenharia Civil - FEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% funcao para a obtençao do vetor de forças e a matriz de rigidez tangente
% para um determinado vetor de deslocamentos
% em relação ao sistema local de coordenadas

function [pb, kb, N]=obtemVetForcasEMatRigBasico(Li,A,mate, epslon)
    % dados de entrada
    % L = comprimento da barra
    % A = area da seção
    % mate = dados do material
    % epslon = deformação da barra no sistema local

    [sigma, Et] = calcTensaoGMP(mate, epslon);

    pb = A*sigma
    kb = (A*Et/Li)
    N = A*sigma


return
```

ANEXO C.11 – função `obtemVetForcasEMatRigEstrutura`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% monta a matrix de rigidez da estrutura

function [Pint, K, N] = obtemVetForcasEMatRigEstrutura(NGLEst, numElems,
coords, conect, dadosElem, gdl, D)

% zeros (numLinhas, numColunas)
K = zeros(NGLEst, NGLEst)
Pint = zeros (NGLEst, 1)

% loop nos elementos
for el = 1: numElems
    el
   % determina os nos do elemento
   noI = conect(el,1)
   noJ = conect(el,2)

   % determina as coordenadas dos nos do elemento
   coordNoI = coords(noI,:)  % copia a linha 'noI' da matriz coords na
matriz coordNoI
   coordNoJ = coords(noJ,:)  % copia a linha 'noJ' da matriz coords na
matriz coordNoJ

   % determina os graus de liberdade do elemento
   grLibElem = obtemGrausLibElemento(noI, noJ, gdl)

   % extrai os deslocamentos das barras, do vetor de deslocamentos da
estrutura
   for i = 1: 4
       i
       gdlEl = grLibElem(i)
       dg(i,1) = D(gdlEl)

   end

   l_dg = size(dg,1)
   c_dg = size(dg,2)

   dgTransp = dg'
   UI = dgTransp(1,1:l_dg/2)
   UJ = dgTransp(1, l_dg/2 + 1:l_dg)

   coordNoIAtual= coordNoI + UI
   coordNoJAtual= coordNoJ + UJ

   % obtem dados da seção e do material
   A = dadosElem(el,1)
   mate = dadosElem(el,2:5)'

   % calcula a matriz de rigidez do elemento em coordenadas globais
   [pg, kg, N]=obtemVetForcasEMatRigGlobal(coordNoI, coordNoJ,
coordNoIAtual, coordNoJAtual, A, mate, dg)
```

```
   % adiciona matrix de rigidez do elemento a matriz de rigidez da
estrutura
   for l = 1: 4
       l
      L = grLibElem(l)
      Pint(L) = Pint(L) + pg(l)

      for c = 1: 4
         C = grLibElem(c)
         K(L, C) = K(L, C) + kg(l,c)
      end
   end

end

%-------------------------------------------------------------------------
-
```

ANEXO C.12 – função `obtemVetForcasEMatRigGlobal`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% funcao para a obtençao do vetor de forças e a matriz de rigidez tangente
% para um determinado vetor de deslocamentos
% em relação ao sistema global de coordenadas

function [pg, kg, N]=obtemVetForcasEMatRigGlobal(coordNoI, coordNoJ,
coordNoIAtual, coordNoJAtual, A, mate, dg)
    %dados de entrada
    % coordNoI = coordenadas do nó I
    % coordNoJ = coordenadas do nó J
    % A = area da seção
    % mate = dados do material
    % dg = vetor de deslocamentos no sistema global

    % dados de saida
    % pg = vetor de forças em relação ao sistema global
    % kg = matriz de rigidez tangente em relaçao ao sistema global

    % obtem o comprimento inicial e os cossenos diretores da barras
        [Li, costi, sinti] = compOrientElem(coordNoI, coordNoJ)

    % obtem o comprimento final e os cossenos diretores da barras
    [Lf, costf, sintf] = compOrientElem(coordNoIAtual, coordNoJAtual)

    % obtem a matriz da rotação da barra
    [T, G] = obtemMatRotacao(costf, sintf)

    % obtem o  deslocamentos no sistema básico

    dL =Lf-Li

    % obtem deformação da barra no sistema básico

    epslon=dL/Li

    % obtem o vetor de forças e a matriz de rigidez tangente
    % em relaçao ao sistema básico
    [pb, kb, N]=obtemVetForcasEMatRigBasico(Li,A,mate, epslon)

    % transforma o vetor de forcas e a matriz de rigidez tangente do
    % sistema básico para o global
    pg = T' * pb
    L=Lf
    KG = (1/L)*G*pb
    KH = T'*kb*T

    kg = KG + KH

return

%-------------------------------------------------------------------------
-
```

ANEXO C.13 – função `PlotaTrajetoriaDeEquilibrio`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

function PlotaTrajetoriaDeEquilibrio(param,carga, Dm)


    no = param.noChave;
    gdl = param.gdlChave;

    for j=1: param.numPassos
       lambda(j)= (j-1)/(param.numPassos-1);
       desloc(j) = Dm(no,gdl,j);
    end

    figure(2)
    hold on
    texto = sprintf('Trajetoria de Equilibrio do no = %d, gdl = %d', no,
gdl);
    title (texto);
    xlabel ('Deslocamento');
    %ylabel ('Lambda');
    ylabel ('Carga');

    plot (desloc, lambda*carga, 'bo-');

return
```

ANEXO C.14 – função `resolveSistEquacoes`

```
% Universidade Federal do Para - UFPa
% Instituto de Tecnologia - ITEC
% Programa de Pós Graduação em Engenharia Civil
% Programa didatico para Analise Não Linear de Trelicas Planas

% resolve o sistema de equacoes

function [P, D] = resolveSistEquacoes(NGLLE, NGLE, K, P, D)

   % particiona o sistema de equacoes
   % sistema original:        K*D = P
   % sistema particionado:    |Kll Klr|  |Dl|   |Pl|
   %                          |       |* |  | = |  |
   %                          |Krl Krr|  |Dr|   |Pr|
   %
   % onde:
   % Dl = deslocamentos nodais desconhecidos
   % Dr = deslocamentos nodais prescritos (conhecidos)
   % Pl = forcas nodais prescritas (conhecidas)
   % Pr = forcas nodais desconhecidas
   %
   % tal que:
   % Kll*Dl + Klr*Dr = Pl          =>    Dl = inv(Kll)*(Pl - Klr*Dr)
   % Krl*Dl + Krr*Dr = Pr          =>    Pr = Krl*Dl + Krr*Dr
   %

   if (NGLLE < NGLE)
      nl = NGLLE
      nt = NGLE

      Kll = K(   1:nl,    1:nl)
      Krl = K(nl+1:nt,    1:nl)
      Klr = K(   1:nl, nl+1:nt)
      Krr = K(nl+1:nt, nl+1:nt)

      Pl  = P(   1:nl, 1)
      Dr  = D(nl+1:nt, 1)

      % resolve o sistema de equacoes particionado
      Dl = inv(Kll)*(Pl - Klr*Dr)
      Pr = Krl*Dl + Krr*Dr

      % monta o vetor completo de deslocamentos nodais
      D = [Dl;
           Dr]

      % monta o vetor completo de forcas nodais
      P = [Pl;
         Pr]

   else
      P = K * D
   end
return

%--------------------------------------------------------------------------
```