ALFREDO GAY NETO
RODRIGO PROVASI CORREIA

AMBIENTE DE ANÁLISE DE RISERS EM ÁGUAS ULTRAPROFUNDAS: ANÁLISE ESTÁTICA, PRÉ E PÓS-PROCESSADOR

Trabalho de formatura dos cursos de graduação dos Departamentos de Engenharia Mecânica e Mecatrônica da Escola Politécnica da Universidade de São Paulo

São Paulo
2006

UNIVERSIDADE DE SÃO PAULO – ESCOLA POLITÉCNICA
DEPARTAMENTOS DE ENGENHARIA MECÂNICA E MECATRÔNICA

Ambiente de Análise de Risers em Águas Ultraprofundas: Análise Estática, Pré e Pós-processador

Alfredo Gay Neto
Rodrigo Provasi Correia

Orientador: Prof. Dr. Clóvis de Arruda Martins

São Paulo
2006

UNIVERSIDADE DE SÃO PAULO – ESCOLA POLITÉCNICA
DEPARTAMENTOS DE ENGENHARIA MECÂNICA E MECATRÔNICA

Ambiente de Análise de Risers em Águas Ultraprofundas: Análise Estática, Pré e Pós-processador

Trabalho de formatura dos cursos de graduação dos Departamentos de Engenharia Mecânica e Mecatrônica da Escola Politécnica da Universidade de São Paulo

Alfredo Gay Neto
Rodrigo Provasi Correia

Orientador: Prof. Dr. Clóvis de Arruda Martins

Áreas de Concentração:
Engenharia Mecânica e Mecatrônica

São Paulo
2006

FICHA CATALOGRÁFICA

Gay Neto, Alfredo
Ambiente de análise de risers em águas ultraprofundas: análise estática, pré e pós-processador / A. Gay Neto, R.P. Correia. -- São Paulo, 2006.
165 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecânica; Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1.Risers 2.Análise estática 3.Método dos elementos finitos I.Correia, Rodrigo Provasi II.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecânica III.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos IV.t.

AGRADECIMENTOS

Agradecemos especialmente a Deus pela oportunidade de realizar esse trabalho, e também a todos que nos ajudaram no decorrer do projeto, desde colegas a professores, todos com colaborações de extrema valia. Especialmente somos gratos ao nosso professor orientador Prof. Dr. Clóvis de Arruda Martins pela experiência passada na solução de problemas, idéias de melhoria de algoritmos, bem como pela supervisão do trabalho e pelo grande incentivo.

Agradecemos, também, à Agência Nacional do Petróleo (ANP) pelo suporte financeiro, através do Programa de Recursos Humanos 19 (PRH 19), que forneceu bolsa auxílio para ambos os autores, bem como adquiriu equipamentos de informática necessários para o desenvolvimento do projeto.

RESUMO

O projeto consiste em desenvolver um ambiente computacional para análise mecânica de risers utilizados na produção de petróleo em águas ultraprofundas, utilizando o método dos elementos finitos. Dentro deste contexto, visam-se mais especificamente, o algoritmo para cálculo de configuração estática, a implementação dos elementos finitos treliça, cabo e pórtico, além das restrições do tipo engaste, articulação e solo. Objetiva-se, fundamentalmente, a obtenção de um código de fácil expansibilidade e portabilidade, o que será conseguido utilizando-se de características como polimorfismo e classes abstratas e também adotando o padrão ANSI para programação, tornando o software multiplataforma. Na primeira etapa foram formulados os elementos de treliça e cabo e as restrições do tipo engaste e articulação, sendo, na etapa final, implementados os elementos tipo pórtico e a restrição solo. O projeto ainda conta com o desenvolvimento de pré e pós-processadores a serem utilizados para facilitar a entrada de dados e visualização de resultados.

ABSTRACT

This project consists in develop a computational workbench to analyze risers' mechanics employed in ultra deepwater oil production. In this context, the objective is to create an algorithm to calculate the static configuration, implement truss, cable and beam finite elements and also fixed, joint and touchdown constraints. One achieved goal was to obtain an easily-expansible and portable code by using polymorphism and abstract classes and also using ANSI specification for programming, leading to platform-independent software. A second stage of the project included a pre and post-processor development. These modules are useful for data input and output, providing perspective visualization and graphic results

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

CAPÍTULO I – CONSIDERAÇÕES INICIAIS

1.1 Introdução

Risers são elementos tubulares que se destinam ao transporte do petróleo produzido em um poço submerso até a unidade de produção flutuante ou, no sentido contrário, do óleo ou do gás já separados da unidade até tubulações submarinas. Até profundidades da ordem de 2000 m, os risers são instalados sob a forma de uma catenária simples ("free-hanging"). Para maiores profundidades são procuradas configurações alternativas, visando minimizar o nível de tração e a fadiga causada pelo movimento induzido pela unidade flutuante ao riser.

Assim, aparecem as configurações com flutuação intermediária, como a "lazy-wave", ou configurações mistas como, por exemplo, o RHAS (riser híbrido auto-sustentável), em que um riser vertical é preso a uma bóia submersa que o traciona e existe um "jumper" flexível que liga o riser vertical à unidade flutuante. Com a demanda crescente de petróleo e com a descoberta de novos campos petrolíferos em águas cada vez mais profundas, surge a necessidade de estudar o comportamento mecânico dessas novas configurações. A Figura 1 mostra algumas possíveis configurações de riser.

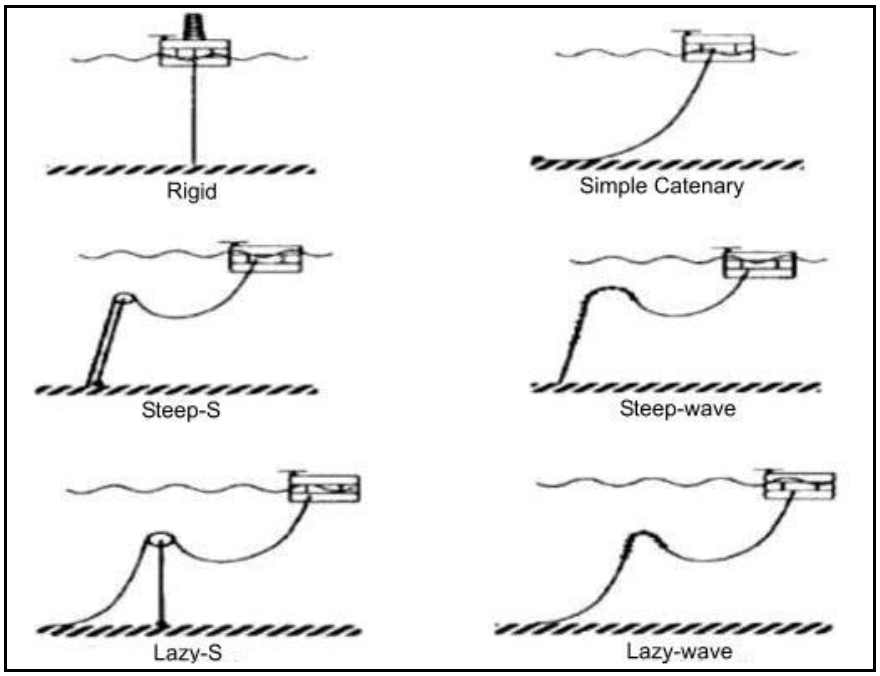


Figura 1 – Algumas Configurações de Riser [1]

Na Escola Politécnica existe uma linha de pesquisa que se dedica há vários anos à análise do comportamento mecânico de risers e outros tipos de cabos submersos com linhas de ancoragem e cabos umbilicais e que já gerou uma série de resultados e trabalhos publicados. O enfoque principal dessa linha de pesquisa tem sido a compreensão clara dos fenômenos físicos envolvidos na mecânica dos cabos submersos e, a partir dessa compreensão, o desenvolvimento de modelos matemáticos simplificados, usando técnicas assintóticas, aproximações lineares e soluções no domínio da freqüência.

Dentro dessa linha de pesquisa existe uma vertente, coordenada pelo orientador, que se dedica à implementação dos modelos desenvolvidos em ferramentas computacionais que visam à análise do comportamento estático e dinâmico dos cabos submersos, tendo gerado diversos produtos: Ristat, Risdin, Poliflex, SteelCat, Utilflex, Poliflex3D e, mais recentemente, Optflex, cada um com uma finalidade específica, mas sempre visando o projeto de risers.

Esse projeto iniciou uma nova etapa, através da criação do programa Ultraflex, em que não se busca mais o desenvolvimento de um programa dedicado a uma finalidade específica, como o projeto de risers para uma gama de configurações particulares, mas o desenvolvimento de um ambiente computacional completo para análise de risers, que seja capaz de tratar as configurações atuais e, ao mesmo tempo, esteja preparado para incluir e testar novas propostas de arranjo físico desses elementos, além de novos materiais.

O ambiente de análise como um todo possui diferentes requisitos para os diferentes módulos, sendo cada requisito descrito com mais detalhes nos capítulos referentes a cada módulo. O módulo central resultante foi construído de tal forma que fosse independente da plataforma computacional, deixando em aberto a possibilidade de paralelização do código, sendo possível ser executado nos ambientes Windows® e Linux®. O pré-processador e o pós-processador não seguiram esse requisito dado que a comunicação entre os módulos é feita através de arquivos XML. Nesses módulos desejou-se um ambiente que fosse amigável para o usuário, permitindo a entrada de dados da forma mais intuitiva possível. Assim, esses módulos foram construídos na plataforma Windows® aproveitando vários de seus recursos gráficos disponíveis, bem como foi feita a implantação de um controle OpenGL® no pré-processador, para que o usuário pudesse visualizar a malha inicial gerada para ser resolvida.

1.2 Motivação

A crescente busca por petróleo em regiões oceânicas de grande profundidade exige, freqüentemente, da engenharia, o desenvolvimento de novas tecnologias que possibilitem a exploração nesses locais e que suportem as condições ambientais severas que podem surgir. Quando se deseja viabilizar a prospecção de petróleo com lâminas d'água de mais de 3000 metros, é necessária a busca por configurações de riser alternativas, que possam se adequar a situações como essas. Para poder prever comportamentos mecânicos e desenvolver essas tecnologias, uma ferramenta genérica de análise estrutural de risers é essencial.

CAPÍTULO II – ANÁLISE ESTÁTICA (CORE)

II.1 Especificações Técnicas

Separaram-se as especificações técnicas do projeto nas duas vertentes apresentadas no estudo de viabilidade: programação de computador e modelagem física e matemática do problema.

Programação de Computador

- O software gerado deve ser multiplataforma;
- A linguagem escolhida deve possibilitar estruturação com orientação a objeto, inclusive recursos de polimorfismo (objetos puramente virtuais);
- Apesar de a velocidade do processamento dos problemas não ser prioridade no projeto, é requerida uma linguagem que dentro de um compromisso com a generalidade do código não desperdice tempo computacional.
- O projeto limita-se, na fase atual, à resolução de problemas estáticos. Porém sua arquitetura de software deve estar aberta para abrigar no futuro outros tipos de análise, como dinâmica.

Modelagem Física e Matemática do Problema

- O modelo físico deve levar em conta não somente esforços de campo gravitacional (peso próprio da estrutura), mas também esforços de interação fluido-estrutura

(correntes marítimas), e deve ser de fácil modificação para a inclusão de outros tipos de componentes como, por exemplo, bóias;

- Devem ser modelados problemas de configuração de riser do tipo catenária (free – hanging), lazy-wave). Além dessas, deve-se estar aberto à resolução de novas configurações futuras, com pouca ou nenhuma mudança de código;
- O modelo matemático deve ser o bastante genérico para permitir a inclusão de novos modelos físicos na expansão futura do projeto.

II.1.1 Linguagem de Programação

Existem diversas opções de linguagem de programação. Algumas, de mais baixo nível, como o Assembler, e outras de mais alto nível, como linguagens interpretadas de softwares comerciais com rotinas prontas para uso, como o MATLAB e SCILAB. Existem ainda algumas linguagens de médio nível, como o C e o C++, que não são interpretadas como o SCILAB e MATLAB, porém para seu uso não é necessário chegar a tão baixo nível, como realizar operações matemáticas bit a bit, que é o caso do Assembler. Existem ainda outras opções, como Java e Object Pascal.

Visando as especificações técnicas mostradas em II.1, escolheu-se a linguagem C++. Em primeiro lugar, dentro do contexto de conhecimento do assunto por parte dos integrantes do projeto, essa escolha demandaria pouco tempo para aprendizado da ferramenta, visto que ambos já conheciam tal linguagem previamente. Além disso, poder-se-ia pensar em utilizar Java, mas essa linguagem, apesar de ser portável e rodar virtualmente em qualquer sistema operacional, apresenta grandes limitações por se tratar de uma linguagem interpretada e não

compilada, o que significa que não é otimizada para a plataforma em questão, nem sempre apresentando o desempenho esperado. Nela se gera apenas um arquivo intermediário, muito dependente do equipamento disponível e da máquina virtual Java para funcionar adequadamente. No caso específico, interessa, também, a generalidade e, em segundo plano, a velocidade de processamento. O Java não proporcionaria mais generalidade do que o C++ em termos de estruturação e encapsulamento, no entanto acarretaria um considerável ônus ao tempo computacional. Pensando-se em outras linguagens, poder-se-ia utilizar Object Pascal, mas esta tem uma forma completamente distinta de C e C++, o que demandaria muito mais tempo, visto que teria que ser aprendida.

Assim C++ foi a escolha natural, destacando ainda que para executar o código em uma nova plataforma, bastaria compilar o mesmo nesse novo ambiente, através da ferramenta de compilação adequada.

As vantagens apresentadas pelo C++ não são só relativas ao desempenho. Por se tratar de uma linguagem orientada a objetos, têm-se recursos que permitem uma melhor estruturação e compreensão do código, além de seu reuso. Conceitos como Herança e Encapsulamento são básicos para a estrutura delineada para o programa. Lembrando que a idéia central deste ambiente é manter flexibilidade com facilidade de expansão e com generalidade suficiente para que a inclusão de novos códigos fique quase que restrita à geração dos próprios, com poucas ou, se possível, nenhuma alteração no restante do programa. Isso pode ser conseguido utilizando o conceito de classes virtuais e herança, ambos presentes na linguagem escolhida.

II.1.2 Modelagem Física e Matemática

A modelagem física deve levar em conta os efeitos apresentados em II.1.Para o problema estático de risers, em geral, nota-se a existência de alta não-linearidade geométrica inerente à forma da estrutura e, em certos casos, a não-linearidade de contato com solo (problemas com TDP – "Touch Down Point").

Quanto aos modelos matemáticos, existem disponíveis na literatura abordagens que se baseiam na integração da equação diferencial do problema para a solução estática. Porém, quando se leva em conta esforços de corrente marítima, por exemplo, não existe mais solução analítica para essa integração.

A fim de priorizar a futura expansibilidade do software, com o mínimo de alterações possíveis, acreditou-se ser necessária a escolha de um modelo matemático mais genérico do que modelos analíticos para determinar a linha elástica do riser, que são particulares para cada tipo de configuração. Um método disponível e que atende perfeitamente aos quesitos apresentados é o Método dos Elementos Finitos (MEF).

Esse método consiste em dividir o problema global a ser estudado em problemas menores de solução local conhecida e fazer com que as condições de contorno de cada um desses sejam adequadas com a física global. Matematicamente, trata-se da união de vários conjuntos, que são domínios de funções locais e que formarão juntos um domínio global, com a união de todas as funções, pelas suas condições de contorno. Trata-se de um método robusto, que utiliza uma formulação fraca, ou seja, garante que as condições matemáticas sejam atendidas na média. Isso se traduz na condição de que os valores da função em pontos específicos – os nós – sejam "exatos" (dentro da formulação matemática). Uma decorrência desta formulação é que se substitui a solução de equações diferenciais parciais por soluções de equações

integrais e que levam à solução de sistemas de matrizes, que possuem ainda algumas características peculiares que, computacionalmente, facilitam muito sua solução.

Apesar de a escolha dos métodos não envolver diretamente a utilização de soluções analíticas, acredita-se que seu estudo é de fundamental importância para que se adquira familiaridade com o problema e com alguns resultados esperados. Podem-se utilizar, ainda, tais soluções para comparação com o método implementado para alguns casos particulares, visando a validação do modelo. Um outro aspecto importante da solução analítica é possibilidade dessa ser utilizada como estimativa inicial da configuração.

II.2 Fundamentação Teórica

O comportamento de um riser, do ponto de vista estático, é muito semelhante ao de um cabo perfeitamente flexível e o efeito da rigidez flexional não é importante, a menos de regiões em torno dos pontos em que há descontinuidade de curvatura no modelo de cabo, como a região do ponto de contato entre o riser e o fundo do mar ("*touchdown point*" – TDP), a conexão do riser à unidade flutuante de produção e os pontos em que há mudança do peso submerso, como na transição entre trechos sem e com flutuação. Este fato, se por um lado permite a adoção de solução analíticas do tipo *boundary-layer*, por outro lado dificulta a solução do problema via métodos numéricos, pois o sistema de equações diferenciais resultantes é muito "rígido".

Para determinar a linha elástica de um riser, é necessário utilizar um modelo não-linear do ponto de vista geométrico, já que os deslocamentos são muito grandes. Além disso, deve-se prever a possibilidade de materiais com comportamento não-linear como o poliuretano, por

exemplo, que é utilizado nos enrijecedores (“bending stiffener”) utilizados para a transição de curvatura no topo do riser. Na região do TDP existe um problema de contato unilateral. O próprio solo em que se apóia o riser tem comportamento não-linear.

Para simular a estática de um riser, face às características apresentadas, não é possível utilizar um código de elementos finitos convencional, tornando necessário o uso de ferramentas específicas. Existem softwares comerciais, desenvolvidos para essa necessidade, mas, como não são abertos, não é possível alterar o seu conteúdo, incluindo novas características.

Para estudar o comportamento de um riser, sob uma configuração genérica, deve se pensar em um modelo de elementos finitos, que leve em conta não-linearidades constitutivas e geométricas, que considere problemas de contato unilateral e que possa ser resolvido no domínio da freqüência e do tempo (no caso de análises dinâmicas).

Para a resolução de análises estáticas, foram desenvolvidos algoritmos iterativos não-lineares, e que levem em conta grandes deslocamentos, mas que considerem o comportamento constitutivo como sendo elástico linear (seguem a Lei de Hooke).

A seguir serão abordados diversos tópicos que construirão o raciocínio do leitor em relação ao que foi efetivamente implementado no software desenvolvido. Inicia-se com uma formulação contínua e analítica de cabos, que é o embasamento para a abordagem discreta do MEF. A seguir, desenvolve-se o MEF linear, seguido do MEF não linear, com seus respectivos algoritmos. A formulação dos elementos modelados também é apresentadas, bem como os carregamentos mais comuns no problema físico do riser. Ainda são apresentadas as condições de contorno implementadas. Por último, uma base de toda a estrutura do código, com grande ênfase na área de programação de computador.

II.2.1 Abordagem Analítica [2]

Nessa seção será apresentado o modelo de cabo mais simples (Figura 2), que resulta em equações diferenciais de equilíbrio para cabos inextensíveis e perfeitamente flexíveis. O único carregamento que será levado em conta é o peso próprio da estrutura, que é equilibrado pela tração que surge no cabo.

Definindo *v* como o peso próprio do cabo por unidade de comprimento, e a força de tração na coordenada S do cabo como *T*, ilustra-se, na Figura 2 o diagrama de corpo livre de um elemento de cabo diferencial.

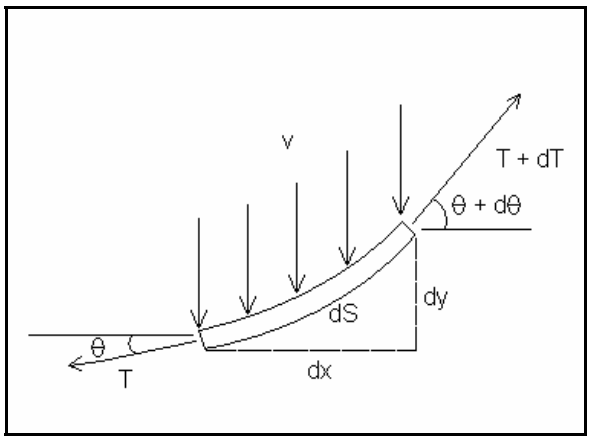


Figura 2 – Elemento de cabo diferencial

Podem ser escritas as seguintes equações que decorrem de relações geométricas:

$$\frac{dx}{dS} = \cos\theta \qquad \frac{dy}{dS} = sen\theta \tag{II.1}$$

Ainda podem ser escritas as equações de equilíbrio estático para o elemento:

- *Direção horizontal – coordenada x:*

$$T\cos\theta = (T + dT)\cos(\theta + d\theta) = T\cos\theta + dT\cos\theta - Tsen\theta d\theta \tag{II.2}$$

Conclui-se que:

$$dT\cos\theta - Tsen\theta d\theta = 0 \Leftrightarrow d(T\cos\theta) = 0 \tag{II.3}$$

Portanto, o valor da força do cabo na direção horizontal possui um valor constante, que será indicado por *H*. Assim, tem-se que:

$$T\cos\theta = H = cte \tag{II.4}$$

- *Direção vertical – coordenada y:*

$$Tsen\theta + vdS = (T + dT)sen(\theta + d\theta) = Tsen\theta + dTsen\theta + T\cos\theta d\theta \tag{II.5}$$

A partir da qual se conclui que:

$$v = \frac{d}{dS}(Tsen\theta) \tag{II.6}$$

Portanto, conclui-se que a componente vertical da força de tração no cabo possui primeira derivada constante em relação à abscissa curvilínea S. Assim, a variação da componente vertical é regida por uma função de primeiro grau.

$$V(S) = vS + C \tag{II.7}$$

Onde C é uma constante, que deve ser determinada a partir de condições de contorno do cabo.

Equilíbrio de momentos:

Sendo o momento devido ao peso próprio um efeito de segunda ordem, pode-se afirmar que as únicas forças que causam momento significativo no cabo são as componentes da tração. Sendo assim:

$$T\cos\theta dy = Tsen\theta dx \tag{II.8}$$

A partir dessas equações de equilíbrio, com uma série de manipulações algébricas pode-se chegar à equação diferencial de equilíbrio – equação (II.9).

$$\frac{d^2y}{dx^2} - \frac{v}{H}\sqrt{1+\left(\frac{dy}{dx}\right)^2} = 0 \tag{II.9}$$

Pode-se mostrar que a função *y(x)* a seguir é solução para a equação (II.9).

$$y(x) = \pm\frac{H}{v}\cosh\left(\frac{v}{H}x + A\right) + B \tag{II.10}$$

Os sinais positivo e negativo da expressão acima bem como as constantes *A* e *B*, são determináveis a partir das condições de contorno do cabo.

Ainda é possível determinar o valor da tração em função da coordenada x, através da expressão dada pela equação (II.11).

$$T = \frac{H}{\cos\left(a\tan\left(senh\left(\frac{v}{H}x + A\right)\right)\right)} \tag{II.11}$$

II.2.2 Abordagem Numérica

Devido às grandes particularidades envolvidas nas soluções analíticas, é interessante realizar uma abordagem mais genérica e que possa facilmente levar em conta efeitos de rigidez axial, flexional e torcional, esforços de um pefil de correntes marítimas, contato unilateral com solo, dentre outros efeitos.

II.2.2.1 Método dos Elementos Finitos Linear

Resumidamente, pode-se afirmar que o MEF linear, quando utilizado para realizar uma análise estrutural estática linear, consiste em realizar os seguintes passos:

I. Baseando-se na geometria de um problema, gerar uma malha formada por elementos com características previamente escolhidas, alocados entre nós e algumas vezes com nós em seu interior, e que sejam representativos em relação ao problema real, ou seja, que consigam descrever com fidelidade a geometria e as condições de contorno do problema físico (Figura 3)

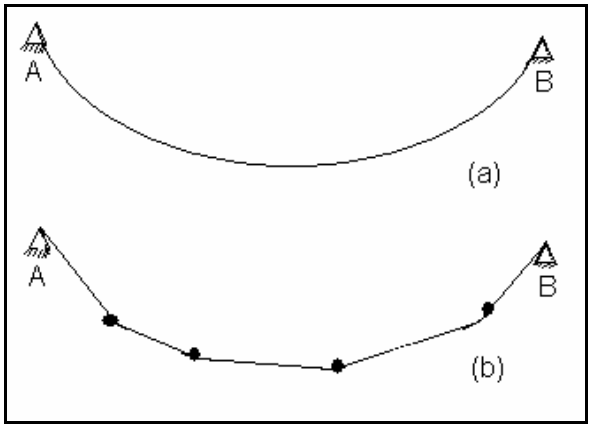


Figura 3 - (a) Um exemplo de configuração de riser (b) Um exemplo de discretização do riser do item (a) através do Método dos Elementos Finitos

II. Calcular a matriz de rigidez de cada um dos elementos individualmente (esse procedimento está detalhado em II.3) e rotacioná-lo para o sistema de coordenadas global (Apêndice A)

III. Através de uma numeração global de graus de liberdade do sistema, montar uma matriz de rigidez global que contenha informações sobre a rigidez de todos os elementos existentes na malha, superpondo informações locais. Essa relação entre numerações global e local envolve uma matriz denominada matriz de conexão;

IV. Resolver um sistema linear que envolve a matriz de rigidez e os esforços nodais, para determinar os deslocamentos em cada grau de liberdade livres;

V. Calcular outras grandezas físicas desejadas para concluir a análise, com base nos valores determinados de deslocamentos nodais. Por exemplo, pode-se desejar saber as tensões de Von Misses em cada nó.

Uma análise envolvendo o MEF envolve prioritariamente a escolha dos tipos de elementos que serão utilizados. Essa é, sem dúvida, alguma uma das partes mais críticas na resolução do problema, visto que é necessário o conhecimento profundo do fenômeno físico que ocorre na estrutura para realizar a escolha adequada do tipo de elemento. O modelo deve levar em conta os aspectos relevantes à análise que será realizada, como os tipos de esforços, tipos de condição de contorno, graus de liberdade existentes, dentre outros aspectos.

- ***Matriz de Conexão***

Uma vez que a abordagem para a resolução utilizando o MEF linear envolve a superposição de efeitos de rigidez local de cada elemento em uma matriz global, que contenha informações

de todos os elementos, é necessária uma organização para montar a matriz global do sistema a partir das matrizes locais dos elementos. Admitindo que o sistema de coordenadas no qual está escrita a matriz local de cada elemento seja o sistema global (as matrizes locais foram rotacionadas de seu sistema local para o global), é possível organizar uma estrutura que relacione uma numeração global de graus de liberdade com uma numeração local. A Figura 4 mostra um exemplo de estrutura de treliças bidimensionais, e em seguida é exibida sua matriz de conexão (Tabela 1).

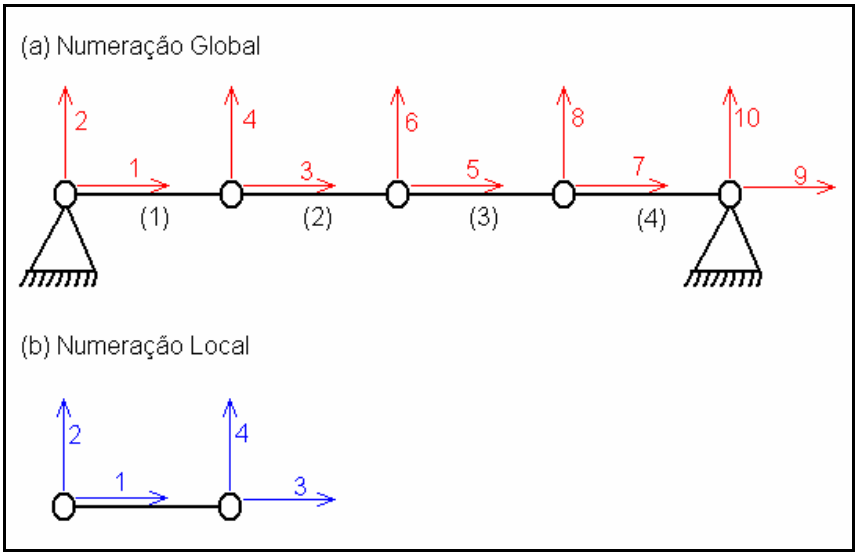


Figura 4 – Exemplo de numeração global (a) e numeração local (b)

Tabela 1 – Conexão entre os graus de liberdade locais e globais

	Elemento	(1)	(2)	(3)	(4)
GL local	*1*	1	3	5	7
	2	2	4	6	8
	3	3	5	7	9
	4	4	6	8	10

Lê-se, por exemplo, para o elemento (3), que seu grau de liberdade 1 local está relacionado com o 5 global, e o grau de liberdade 2 local está relacionado com o 6 global, e assim sucessivamente.

Assim sendo, é possível montar a matriz de rigidez global superpondo as matrizes locais, relacionando cada posição local com as posições globais contidas na matriz de conexão. Portanto, a matriz de conexão estabelece uma relação entre a numeração dos graus de liberdade locais e globais.

- ***Condensação Estática* [3]**

Uma característica intrínseca a problemas estruturais é a presença de condições de contorno. Essas determinam quais os graus de liberdade dos elementos dispostos na malha que estarão livres e quais estarão fixos pelas restrições.

Na prática para a resolução de problemas de MEF é comum atribuir uma numeração global a todos os graus de liberdade existentes nos elementos, ainda que estejam restritos pelas condições de contorno e, portanto, não possuam movimento livre. Isso é feito para facilitar o cálculo de reações nos vínculos, visto que esses "graus de liberdade" fixados são inclusos na matriz de rigidez global e, dependendo da maneira que é feita sua numeração, é possível dispor a matriz da seguinte maneira:

$$\mathbf{K_G} = \begin{bmatrix} \mathbf{K_{AA}} & \mathbf{K_{AB}} \\ \mathbf{K_{BA}} & \mathbf{K_{BB}} \end{bmatrix} \tag{II.12}$$

Onde: $\mathbf{K_G}$ é a matriz de rigidez global do sistema;

$\mathbf{K_{AA}}$ é a parte da matriz $\mathbf{K_G}$ composta pelos coeficientes de influência de rigidez provenientes apenas da interação graus de liberdade livres;

$\mathbf{K_{BB}}$ é a parte da matriz $\mathbf{K_G}$ composta pelos coeficientes de influência de rigidez provenientes apenas da interação de graus de liberdade fixados pelas condições de contorno;

$\mathbf{K_{AB}}$ e $\mathbf{K_{BA}}$ são matrizes compostas por coeficientes de influência de rigidez provenientes da interação entre graus de liberdade livres e graus de liberdade fixados pelas condições de contorno.

Escrevendo a equação de equilíbrio de esforços para um problema estático na forma matricial, tem-se:

$$\mathbf{K_G u} = \mathbf{f} \tag{II.13}$$

Onde: $\mathbf{K_G}$ é a matriz de rigidez global do sistema;

$\mathbf{f}$ é o vetor de esforços nos graus de liberdade do sistema;

$\mathbf{u}$ é o vetor de deslocamentos dos graus de liberdade do sistema.

Portanto, é possível descrever um problema estático linear com a seguinte equação:

$$\begin{bmatrix} \mathbf{K_{AA}} & \mathbf{K_{AB}} \\ \mathbf{K_{BA}} & \mathbf{K_{BB}} \end{bmatrix} \begin{bmatrix} \mathbf{u_A} \\ \mathbf{u_B} \end{bmatrix} = \begin{bmatrix} \mathbf{f_A} \\ \mathbf{f_B} \end{bmatrix} \tag{II.14}$$

Note que é realizada uma separação do vetor u em duas partes $\mathbf{u_A}$ e $\mathbf{u_B}$, que representam respectivamente os deslocamentos dos graus de liberdade livres e os deslocamentos dos graus de liberdade fixados pelas condições de contorno. Um procedimento análogo é feito com o vetor de esforços. Assim, podem ser escritas duas equações que efetivamente resolverão o problema – equação (II.15).

$$\begin{cases} \mathbf{K}_{AA}\mathbf{u}_A + \mathbf{K}_{AB}\mathbf{u}_B = \mathbf{f}_A \\ \mathbf{K}_{BA}\mathbf{u}_A + \mathbf{K}_{BB}\mathbf{u}_B = \mathbf{f}_B \end{cases} \tag{II.15}$$

Nessas equações existem dois vetores que são incógnitas: $\mathbf{u}_A$ e $\mathbf{f}_B$. O vetor $\mathbf{u}_B$ permite incluir na análise a imposição de deslocamentos para certos graus de liberdade, e o vetor $\mathbf{f}_A$ contém o carregamento imposto na estrutura. Para determinar o vetor de deslocamentos livres da primeira equação, é necessária a resolução do sistema linear em $\mathbf{u}_A$ da equação (II.16).

$$\mathbf{K}_{AA}\mathbf{u}_A = \left(\mathbf{f}_A - \mathbf{K}_{AB}\mathbf{u}_B\right) \tag{II.16}$$

Uma vez determinado $\mathbf{u}_A$, é possível determinar o vetor $\mathbf{f}_B$ através da segunda equação. Dessa forma é concluída a análise linear da ação do carregamento na estrutura.

A técnica utilizada para dividir o problema e as equações matriciais em equações que envolvem sub-matrizes conforme mostrado anteriormente, é denominada Condensação Estática.

- ***Método Para a Resolução de Sistemas Lineares***

A equação (II.16) ilustra a necessidade de um método de resolução de sistemas lineares para determinação do vetor $\mathbf{u}_A$. Em MEF geralmente é possível arranjar a matriz de rigidez global na forma de banda, devido à forma como ocorre a interconexão entre os elementos. Essa estrutura permite o armazenamento com menos alocação de memória, e certa otimização na resolução do sistema linear.

Dois métodos de fatoração matricial foram aplicados no software desenvolvido para a resolução de sistemas lineares. Serão, a seguir, descritos ambos os métodos.

- *Fatoração Cholesky* [4]

O algoritmo de Cholesky implementado pressupõe que irá ser resolvido um sistema linear do tipo $\mathbf{Ax} = \mathbf{b}$, com algumas hipóteses:

As dimensões são compatíveis – característica básica para qualquer sistema de equações lineares.

A matriz A é positiva definida, o que significa afirmar que, dada a matriz e um vetor **v** qualquer, $\begin{cases} \mathbf{v}^{\mathbf{T}}\mathbf{Av} \geq \mathbf{0} \\ \mathbf{v}^{\mathbf{T}}\mathbf{Av} = \mathbf{0} \Leftrightarrow \mathbf{v} = \mathbf{0} \end{cases}$.

Como se verificam sempre as condições de entrada do problema e como a matriz tem que ser simétrica e definida positiva, condições que são satisfeitas pela matriz de rigidez do problema, então o método pode ser utilizado.

O método consiste em obter uma matriz triangular inferior **L** tal que:

$$\mathbf{LL}^{\mathbf{T}} = \mathbf{A} \tag{II.17}$$

Sendo $\mathbf{L}^{\mathbf{T}}$ a matriz transposta de **L**.

O método para a determinação dos elementos de **L** é feito através do Algoritmo de Crout, que resolve um conjunto de $n + n^2$ equações de uma maneira simplificada, apenas arranjando as matrizes numa ordem específica, onde n é a dimensão da matriz **A**.

A decomposição consiste nos seguintes cálculos:

$$l_{ii} = \left(a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2 \right)^{1/2} \tag{II.18}$$

$$l_{ij} = \frac{1}{l_{ii}}\left(a_{ij} - \sum_{k=1}^{i-1} l_{ik} l_{jk} \right) \qquad \text{(II.19)}$$

Onde $i= 1, 2,..., n$ e $j = i+1, i+2,... , n$.

O único detalhe a ser acrescentado nessa formulação, quanto à sua adaptação para a forma de banda da matriz, é o fato de ser verificada a diferença entre os índices *i, j e k* em cada operação, uma vez que esses podem estar situados fora da banda da matriz. Nesse caso, a posição da matriz é nula.

Quanto à alocação de memória, durante a fatoração a matriz **L** é alocada em tempo real no mesmo espaço de memória do qual são lidos os termos da matriz **A**. Isso provê economia de memória durante o processamento.

Após a decomposição da matriz **A** através do método de Cholesky, parte-se para a resolução do sistema, que se torna trivial, já que é solucionado por retro substituição. Resolve-se o sistema da seguinte forma:

$$\mathbf{Ax} = \mathbf{b} \qquad \text{(II.20)}$$

Mas $\mathbf{A} = \mathbf{LL}^{\mathbf{T}}$. Logo:

$$\mathbf{LL}^{\mathbf{T}}\mathbf{x} = \mathbf{b} \qquad \text{(II.21)}$$

Fazendo $\mathbf{y} = \mathbf{L}^{\mathbf{T}}\mathbf{x}$, temos:

$$\mathbf{Ly} = \mathbf{b} \qquad \text{(II.22)}$$

Como tanto **L** e **b** são conhecidos e **L** é uma matriz triangular inferior, resolvem-se as equações sequencialmente de cima para baixo. Repete-se o procedimento, fazendo:

$$\mathbf{L}^{\mathbf{T}}\mathbf{x} = \mathbf{y} \qquad \text{(II.23)}$$

Já que agora tanto **L** e **y** são conhecidos. Assim, por substituição reversa, encontram-se os valores de **x** desejados.

- *Fatoração* $\mathbf{LDL^T}$ [5]

Dado um sistema linear da forma **Ax=b**, pode-se decompor a matriz **A** da seguinte forma:

$$\mathbf{A} = \mathbf{LDL^T} \tag{II.24}$$

Onde **L** é uma matriz quadrada triangular inferior e **D** é uma matriz diagonal. Denotando por l_{ij} e d_{ij} os coeficientes da matriz **L** e **D** respectivamente, pode-se calculá-los da seguinte forma:

$$d_j = a_{jj} - \sum_{k=1}^{j-1} d_k (l_{jk}) \tag{II.25}$$

$$l_{ij} = \frac{1}{d_j}\left(a_{ij} - \sum_{k=1}^{j-1} d_k l_{jk} l_{ik} \right) \tag{II.26}$$

Com $i = j+1, ..., n$ e $j = 1, ..., n$, onde n é a ordem da matriz **A**. Os valores dos somatórios são nulos quando o limite inferior supera o limite superior.

Voltando ao sistema original e substituindo a matriz **A** fatorada, tem-se:

$$\mathbf{LDL^T x} = \mathbf{b} \tag{II.27}$$

Substituindo o sistema acima por três sistemas de resolução simplificada, têm-se:

$$\mathbf{L^T x} = \mathbf{y} \tag{II.28}$$

$$\mathbf{Dy} = \mathbf{z} \tag{II.29}$$

$$\mathbf{Lz} = \mathbf{b} \tag{II.30}$$

Resolvendo os três sistemas acima por retro substituição, na ordem em que aparecem suas equações, determina-se o valor do vetor **x**.

Quanto ao gerenciamento de memória, o método goza das mesmas vantagens da Fatoração Cholesky.

II.2.2.2 Método dos Elementos Finitos Não-linear

Dentro do contexto da análise estrutural, o MEF não linear é necessário para a resolução de problemas nos quais ocorrem grandes deslocamentos, grandes deformações ou restrições não lineares. Para o problema específico de uma configuração estática de riser, como não se sabe previamente a posição de equilíbrio no início, podem ocorrer grandes deslocamentos na malha inicial até que a mesma atinja o equilíbrio, uma vez que essa não necessariamente está próxima da solução. Essa grande mudança de geometria causa a variação da matriz de rigidez do sistema, outrora tratada como constante para problemas lineares. Daí vem a não linearidade geométrica.

Se a formulação do elemento levar em conta algum efeito constitutivo de deformação plástica, também ocorrerá uma variação na matriz de rigidez do sistema, mesmo que não haja grandes deslocamentos, mas haja grandes deformações.

Ainda no caso em que se aplica uma condição de contorno, como por exemplo, a imposição de um solo horizontal, podem ocorrer grandes deslocamentos nodais, acarretando também em não linearidade geométrica.

Um método não linear consagrado para uso em análise estrutural é o Método de Newton-Raphson. Muito utilizado por apresentar rápida taxa de convergência (quadrática), o Método

de Newton possui um simples interpretação geométrica para um caso de função de uma variável. A Figura 5 ilustra como ocorre a convergência, a partir de uma estimativa inicial de raiz x_1, e refinando-se o valor dessa estimativa, sempre pela aproximação da função por sua tangente em cada ponto. O método, portanto, necessita do cálculo da derivada da função em todos os pontos (x_1, x_2, x_3,...) que forem aproximações da raiz, durante a convergência.

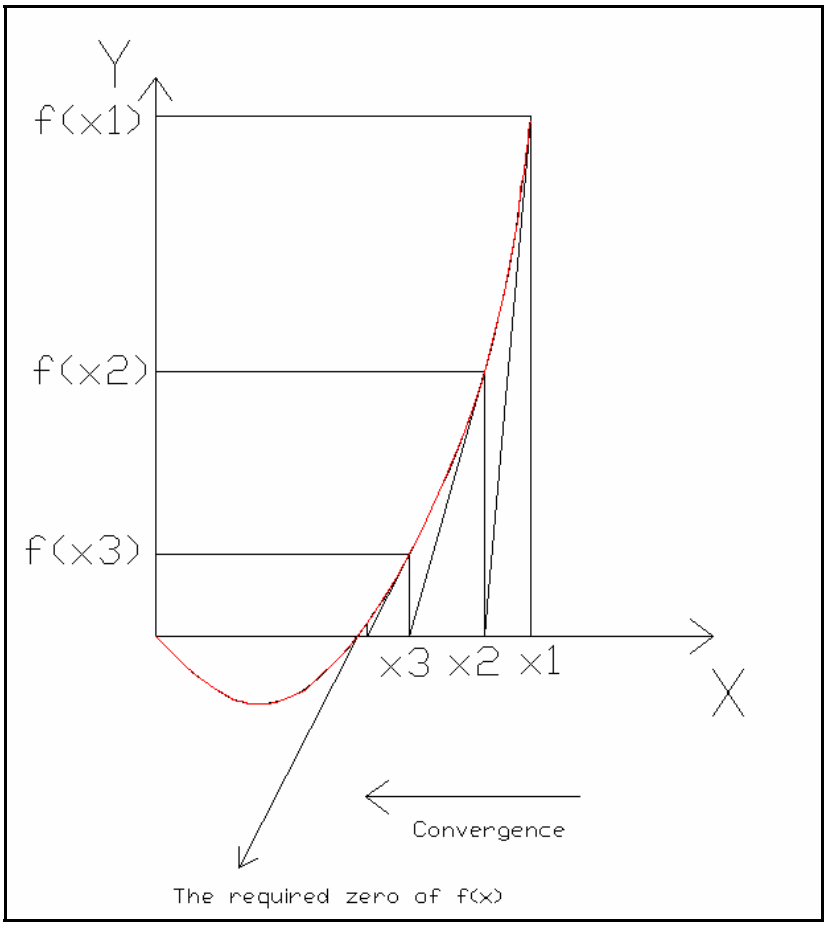


Figura 5 – Interpretação Geométrica para o Método de *Newton-Raphson*

Como pode ser observado na Figura 5 o Método de Newton-Rapshon utiliza-se a cada iteração de uma aproximação linear (tangente) para a função.

É possível generalizar o Método para um problema n – dimensional. Nesse caso, continua valendo o raciocínio de aproximar uma função não linear em funções lineares a cada iteração.

Por isso foi abordado o item II.2.2.1, pois a cada iteração da resolução do problema não linear, será feita uma análise linear completa, sendo a linearização feita em torno da geometria da malha nessa iteração. A seguir está desenvolvida a generalização do Método de Newton-Raphson.

- **Método de Newton-Raphson *para um espaço* n-*dimensional* [6]**

Dada uma função do vetor g(u) de *n* variáveis, situada em um espaço de ordem *n* ($\mathbf{g}:\Re^n \to \Re^n$), pode-se escrever a expansão de **g(u)** utilizando-se a série de Taylor, em torno de um ponto genericamente chamado de $\mathbf{u_i}$.

$$\mathbf{g(u)} = \mathbf{g(u_i)} + \left.\frac{\mathbf{dg}}{\mathbf{du}}\right|_{\mathbf{u_i}} \left(\mathbf{u}-\mathbf{u_i}\right) + \frac{1}{2!}\left.\frac{\mathbf{d^2 g}}{\mathbf{du^2}}\right|_{\mathbf{u_i}} \left(\mathbf{u}-\mathbf{u_i}\right)^2 + \frac{1}{3!}\left.\frac{\mathbf{d^3 g}}{\mathbf{du^3}}\right|_{\mathbf{u_i}} \left(\mathbf{u}-\mathbf{u_i}\right)^3 + \mathbf{O}^4 \qquad \text{(II.31)}$$

Pode-se denotar por **g(u*)** um vetor identicamente nulo. Assim, **u*** representa o vetor que leva a função a seu valor nulo, e, portanto, representa a raiz da função.

O *Método de Newton* para um espaço *n*-dimensional se propõe a apresentar uma metodologia para calcular, iterativamente, o valor do vetor **u***. Para conseguir uma fórmula de recorrência, simplifica-se a expansão de Taylor para primeira ordem, e impõe-se que o valor de $\mathbf{g(u_{i+1})}$ seja identicamente nulo.

$$\mathbf{g(u_i)} + \left.\frac{d\mathbf{g}}{d\mathbf{u}}\right|_{\mathbf{u_i}} \left(\mathbf{u_{i+1}}-\mathbf{u_i}\right) = \mathbf{0} \qquad \text{(II.32)}$$

Na expressão acima o vetor $\mathbf{u_{i+1}}$ representa uma aproximação de primeira ordem para o zero da função **g(u)**. É possível determiná-lo, desde que se conheçam os valores das componentes

do vetor $\mathbf{u_i}$ em torno do qual se está expandindo a série, bem como o valor da derivada $\left.\frac{d\mathbf{g}}{d\mathbf{u}}\right|_{\mathbf{u_i}}$, que é uma matriz *n* x *n*.

Se for tomada uma nova expansão de Taylor em torno do ponto $\mathbf{u_{i+1}}$ recém determinado, recalculando-se o novo valor da derivada $\frac{d\mathbf{g}}{d\mathbf{u}}$, desta vez em torno do ponto $\mathbf{u_{i+1}}$, é possível calcular uma aproximação mais precisa para o vetor **u***.

Generalizando, pode-se escrever a fórmula de recorrência para o cálculo da aproximação de **u***, como mostrado na equação (II.33).

$$\mathbf{u_{i+1}} = \mathbf{u_i} - \left(\left.\frac{d\mathbf{g}}{d\mathbf{u}}\right|_{\mathbf{u_i}} \right)^{-1} \mathbf{g(u_i)} \tag{II.33}$$

Chamando o valor da derivada $\left.\frac{d\mathbf{g}}{d\mathbf{u}}\right|_{\mathbf{u_i}}$ de $\mathbf{K_T}$, e definindo um vetor:

$$\mathbf{\Delta u} = \mathbf{u_{i+1}} - \mathbf{u_i} \tag{II.34}$$

Pode-se escrever a fórmula de recorrência de outra forma, mais prática do ponto de vista computacional:

$$\begin{aligned} \mathbf{K_{T_i}\, \Delta u} &= \mathbf{g(u_i)} \\ \mathbf{u_{i+1}} &= \mathbf{u_i} + \mathbf{\Delta u} \end{aligned} \tag{II.35}$$

Utilizando-se essa formulação, primeiro deve ser calculado o valor de $\mathbf{\Delta u}$ através da resolução de um sistema linear, por exemplo, e depois se deve calcular o valor da nova aproximação da raiz da função ($\mathbf{u_{i+1}}$), incrementando-se o valor da aproximação anterior ($\mathbf{u_i}$) com o vetor $\mathbf{\Delta u}$.

- ***Aplicação do* Método de Newton-Raphson *para o MEF***

Como foi visto no item anterior, o método de Newton-Raphson pode ser generalizado para um espaço *n*-dimensional. No entanto, sua aplicação envolve uma função, cuja raiz se deseja determinar.

Para aplicar o Método de Newton-Raphson para um problema estrutural discretizado com elementos finitos, define-se a seguinte função:

$$\mathbf{g(u)} = \mathbf{r} - \mathbf{f} \tag{II.36}$$

Onde: **u** é o vetor de deslocamentos dos graus de liberdade livres do sistema

f é o vetor de esforços externos (carregamentos) aplicados em cada grau de liberdade livre do sistema

r é o vetor dos esforços restauradores (internos) aplicados aos graus de liberdade livres do sistema

g(u) é chamada de função de esforços desbalanceados

Abaixo é feita uma interpretação dos termos que aparecem na fórmula de recorrência do Método de Newton-Raphson, na aplicação para o MEF não linear:

$\mathbf{K}_{\mathbf{T_i}} = \left.\frac{d\mathbf{g}}{d\mathbf{u}}\right|_{\mathbf{u_i}}$ é a derivada da função de esforços desbalanceados em relação ao vetor de deslocamentos u. O resultado será a matriz de rigidez do sistema, a qual deve ser recalculada a cada iteração. Trata-se, portanto, da rigidez tangente;

Δu é o vetor de deslocamentos que ocorre a cada iteração. É o resultado direto da resolução de um sistema linear, a cada iteração;

$\mathbf{u_i}$ é o vetor que descreve o total de deslocamentos em cada grau de liberdade ocorrido até a iteração *i*. Descreve a posição geométrica de todo o sistema, a partir das coordenada da malha inicial;

$\mathbf{u_{i+1}}$ é o resultado da soma $\mathbf{u_i} + \mathbf{\Delta u}$, e fisicamente representa o novo deslocamento total em cada grau de liberdade, após a resolução de cada iteração do sistema. Será o $\mathbf{u_i}$ da próxima iteração.

- ***Matriz de rigidez de um elemento – Interpretações físicas*** **[6]**

Calcular a matriz de rigidez de um elemento se resume a calcular a derivada da função de esforços desbalanceados em relação aos deslocamentos, em cada grau de liberdade do elemento.

Para formalizar este cálculo, serão definidos dois novos entes matemáticos, em função dos quais é possível escrever o vetor de esforços restauradores no sistema *global* de coordenadas. É possível escrever:

$$\mathbf{r} = \mathbf{C}\mathbf{n} \qquad \text{(II.37)}$$

Onde: **n** - representa o vetor de esforços restauradores da estrutura, escritos no sistema local do elemento;

C - representa a matriz de rotação, que leva os esforços do sistema local de cada elemento, para o sistema global de coordenadas.

Assim, a função $\mathbf{g(u)}$ pode ser escrita por:

$$\mathbf{g(u)} = \mathbf{Cn} - \mathbf{f} \tag{II.38}$$

Derivando a expressão da função em relação a **u**, tem-se a equação (6).

$$\mathbf{K_T} = \frac{d\mathbf{g}}{d\mathbf{u}} = \mathbf{C}\frac{d\mathbf{n}}{d\mathbf{u}} + \mathbf{n}^{\mathbf{T}}\frac{d\mathbf{C}}{d\mathbf{u}} - \frac{d\mathbf{f}}{d\mathbf{u}} \tag{II.39}$$

A interpretação física de cada termo é a seguinte:

$\mathbf{C}\frac{d\mathbf{n}}{d\mathbf{u}}$ é a rigidez constitutiva do elemento, dada sua geometria constante;

$\mathbf{n}^{\mathbf{T}}\frac{d\mathbf{C}}{d\mathbf{u}}$ é a rigidez geométrica da estrutura, dado um estado de tensões constantes;

$\frac{d\mathbf{f}}{d\mathbf{u}}$ é a rigidez dos esforços externos, e está associada à mudança do valor do carregamento, em função dos deslocamentos sofridos pelo elemento.

Dessa forma, para cada elemento é possível dividir os efeitos de rigidez *interna* em duas componentes: rigidez constitutiva e rigidez geométrica.

- ***Cálculo dos esforços restauradores de cada elemento***

Como mostrado no procedimento de aplicação do MEF não linear, é necessário o cálculo dos esforços restauradores de cada elemento, frente aos deslocamentos que ocorrem nos seus graus de liberdade. Esses esforços são utilizados, juntamente com o carregamento externo, para a construção da função dos esforços desbalanceados.

Foram desenvolvidas duas maneiras para cálculo desses esforços restauradores:

A. Utilizando a rigidez tangente

B. Utilizando as funções de forma de cada elemento

- *Método A*

O método A é de fácil expansibilidade e adaptação para novos elementos a serem criados no futuro, envolvendo um cálculo matricial aproximado dos esforços acumulados devido à aplicação de deslocamentos nos graus de liberdade do sistema. Conhecendo a matriz de rigidez dos elementos, e os deslocamentos em cada grau de liberdade, é possível realizar esse cálculo. A Figura 6a ilustra o comportamento de um grau de liberdade (u) que apresenta uma curva de rigidez não linear. Deseja-se aplicar, por exemplo, um esforço F nesse grau de liberdade. O valor K_1 representa a rigidez tangente com deslocamento nulo. Pode-se calcular, com base em K_1, um valor de deslocamento u_1, portanto, através de uma análise linear. É necessário calcular o valor de F_a (esforço restaurador da estrutura), para calcular o esforço desbalanceado, ilustrado na figura por (F-F_a). Através da Figura 6b, é possível visualizar que o cálculo exato de F_a deve ser feito através do deslocamento u_1 determinado anteriormente e de uma rigidez K_s, chamada de rigidez secante, a qual não é conhecida.

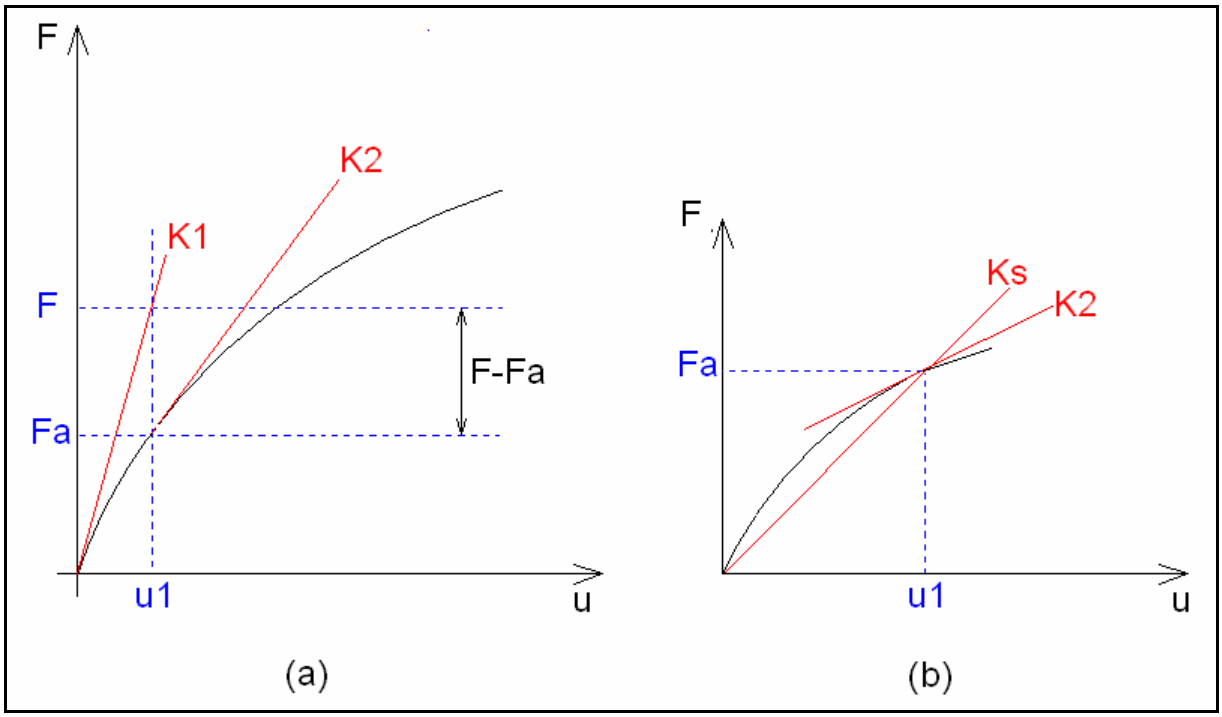


Figura 6 – (a) Rigidez tangente de um grau de liberdade (b) Detalhe da curva

O resultado seria:

$$F_a = K_s u_1 \tag{II.40}$$

Se for tomada a aproximação de que a nova rigidez da estrutura (K_2) após sofrer o deslocamento u_1 é muito próxima de K_s, é possível calcular os esforços F_a de maneira aproximada:

$$F_a \approx K_2 u_1 \tag{II.41}$$

Essa maneira aproximada de calcular F_a, leva a um cálculo de função esforço desbalanceado, para esse grau de liberdade ilustrado em (II.42).

$$g(u) = F - F_a \tag{II.42}$$

O método apresentado apresenta bons resultados para pequenos valores de carregamento F. Quando se deseja aplicar grandes valores de F, é possível dividir o carregamento em pequenos incrementos, tão pequenos quanto se queiram. Portanto, o método pode ser usado de maneira incremental para cálculos com alta não linearidade.

- *Método B*

Esse método consiste em um cálculo específico, diferente para cada elemento, envolvendo particularidades de tipos de esforços envolvidos em suas formulações individuais.

Pode-se apresentar nesse item a metodologia geral que é feita, porém detalhes só podem ser observados em cada tipo de elemento. O método consiste nas seguintes etapas:

- Utilizando as funções de forma e os deslocamentos (escritos no sistema de coordenadas local) de cada grau de liberdade do elemento, obtém-se uma curva parametrizada que representa a linha elástica do elemento.

- A linha elástica obtida representa a configuração deformada do elemento, e através dela, pode-se utilizar expressões geométricas e cinemáticas para calcular os esforços que surgem em cada grau de liberdade para manter a linha elástica obtida.

- Os valores de esforços calculados são os esforços restauradores do elemento, e são matematicamente exatos dentro da formulação utilizada.

Cada tipo de elemento possui uma seção que indicará quais as equações utilizadas para o cálculo de esforços restauradores através desse método.

II.3 Elementos

II.3.1 Elemento tipo Treliça – "Truss"

O elemento do tipo treliça, como esquematizado na Figura 7 possui seis graus de liberdade, sendo todos eles de translação. Graus de liberdade de rotação não são contemplados, e, portanto não necessariamente existe continuidade de ângulo de inclinação entre esses elementos. A continuidade ângulo de inclinação é garantida somente no interior de cada elemento, por sua função de forma.

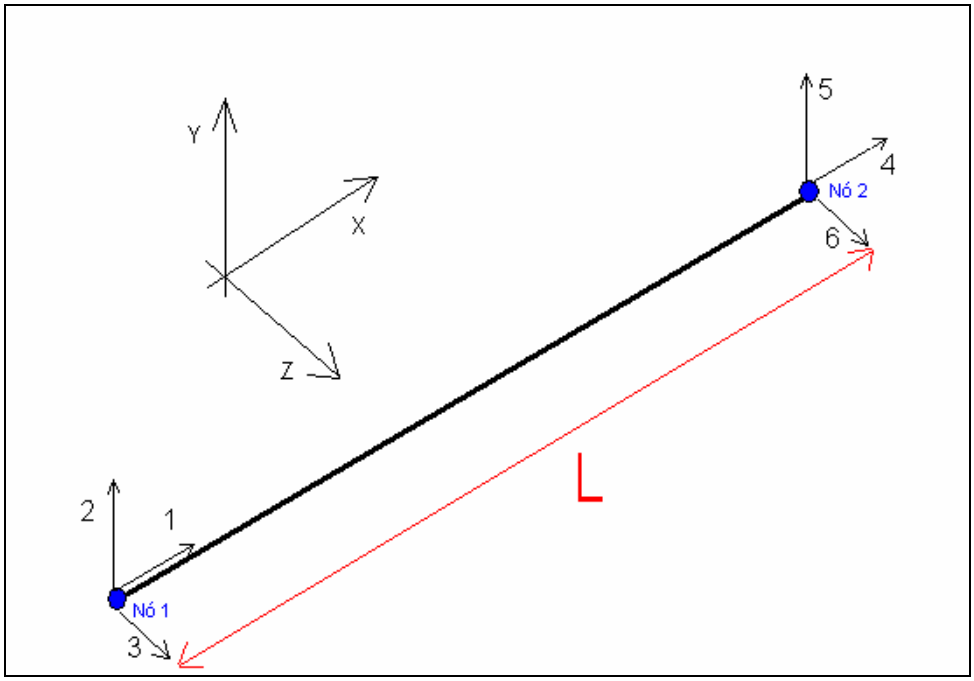


Figura 7 – Elemento do Tipo Treliça com 6 graus de liberdade

Os deslocamentos para cada ponto dessa estrutura estão parametrizados em função do parâmetro t, e são dados pelas equações (II.43), (II.44) e (II.45), onde δ_i e φ_i indicam o

deslocamento e a função de forma para o grau de liberdade *i*, respectivamente. Assim escreve-se as coordenadas *x, y* e *z* do elemento em função de *t*.

$$x(t)= \varphi_1(t)\delta_1 + \varphi_4(t)\delta_4 + t \tag{II.43}$$

$$y(t)= \varphi_2(t)\delta_2 + \varphi_5(t)\delta_5 \tag{II.44}$$

$$w(t)= \varphi_3(t)\delta_3 + \varphi_6(t)\delta_6 \tag{II.45}$$

Assim, pode-se escrever a curva parametrizada - equação (II.46) - desse elemento também em função do parâmetro *t*, que pode variar entre 0 e *L* (sendo L o comprimento inicial do elemento).

$$\Gamma(t) = \big(x(t), y(t), z(t)\big) \tag{II.46}$$

- ***Determinação das Funções de Forma***

As funções de forma têm o aspecto mostrado na equação (II.47), onde *a* e *b* são variáveis a serem determinadas a partir das condições de contorno para cada grau de liberdade.

$$\varphi(t) = at + b \tag{II.47}$$

A determinação das constantes *a* e *b* é feita impondo-se um deslocamento unitário na direção do grau de liberdade em questão e posteriormente, através das condições de contorno impostas por este deslocamento, determinam-se as constantes e conseqüentemente a função de forma para o respectivo grau de liberdade. A Tabela 2 mostra as condições de contorno e funções de forma obtidas para cada grau de liberdade do elemento.

Tabela 2 – Funções de Forma para o elemento de Treliça

Grau de liberdade 1	**Grau de liberdade 2**	**Grau de liberdade 3**
$\varphi_1(0)=1 \quad \varphi_1(L)=0$	$\varphi_2(0)=1 \quad \varphi_2(L)=0$	$\varphi_3(0)=1 \quad \varphi_3(L)=0$
$\varphi_1(t)=1-\frac{t}{L}$	$\varphi_2(t)=1-\frac{t}{L}$	$\varphi_3(t)=1-\frac{t}{L}$
Grau de liberdade 4	**Grau de liberdade 5**	**Grau de liberdade 6**
$\varphi_4(0)=0 \quad \varphi_4(L)=1$	$\varphi_5(0)=0 \quad \varphi_5(L)=1$	$\varphi_6(0)=0 \quad \varphi_6(L)=1$
$\varphi_4(t)=\frac{t}{L}$	$\varphi_5(t)=\frac{t}{L}$	$\varphi_6(t)=\frac{t}{L}$

Substituindo nas equações (II.43), (II.44) e (II.45) as funções de forma já calculadas, pode-se escrever as equações (II.48), (II.49) e (II.50).

$$x(t)=\left(1-\frac{t}{L}\right)\cdot\delta_1+\left(\frac{t}{L}\right)\cdot\delta_4+t \tag{II.48}$$

$$y(t)=\left(1-\frac{t}{L}\right)\cdot\delta_2+\left(\frac{t}{L}\right)\cdot\delta_5 \tag{II.49}$$

$$z(t)=\left(1-\frac{t}{L}\right)\cdot\delta_3+\left(\frac{t}{L}\right)\cdot\delta_6 \tag{II.50}$$

- ***Cálculo do comprimento da barra na configuração deformada***

O novo comprimento na configuração deformada pode ser calculado através da equação:

$$L'=\int_0^L\left|\frac{d\Gamma}{dt}\right|dt \tag{II.51}$$

Desenvolvendo o integrando, obtém-se:

$$\left|\frac{d\Gamma}{dt}\right| = \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2 + \left(\frac{dz}{dt}\right)^2} = \sqrt{\left(\frac{\delta_4 - \delta_1}{L} + 1\right)^2 + \frac{(\delta_5 - \delta_2)^2}{L^2} + \frac{(\delta_6 - \delta_3)^2}{L^2}} = \\ = \frac{1}{L}\sqrt{(\delta_4 - \delta_1)^2 + L^2 + 2L(\delta_4 - \delta_1) + (\delta_5 - \delta_2)^2 + (\delta_6 - \delta_3)^2} \quad \text{(II.52)}$$

Portanto, a integral fica bastante simplificada, e o valor de L' pode ser calculado como mostrado na equação (II.53).

$$L' = \sqrt{(\delta_4 - \delta_1)^2 + L^2 + 2L(\delta_4 - \delta_1) + (\delta_5 - \delta_2)^2 + (\delta_6 - \delta_3)^2} \quad \text{(II.53)}$$

- ***Cálculo da constante elástica para a direção da barra de treliça***

Admitindo uma força de magnitude *T* na direção axial, um deslocamento δ e uma constante elástica de proporcionalidade *k* entre *T* e δ tem-se:

$$T = k\delta \Leftrightarrow \delta = \frac{T}{k} \quad \text{(II.54)}$$

O valor do trabalho *W* realizado pela força *T* é numericamente igual ao trabalho complementar, admitindo-se que o material é linear elástico. Assim:

$$W = W^* = \int_0^T \delta dT = \int_0^T \frac{T}{k} dT = \frac{T^2}{2k} \quad \text{(II.55)}$$

A energia de deformação por unidade de volume da treliça (*u*) também possui mesmo valor que a energia complementar, dado que o material é elástico linear. Dessa forma, pode-se escrever que:

$$u = u^* = \int_0^\varepsilon \sigma d\varepsilon = \int_0^\sigma \varepsilon d\sigma = \int_0^\sigma \frac{\sigma}{E} d\sigma = \frac{\sigma^2}{2E} \quad \text{(II.56)}$$

Calculando agora a energia de deformação (U), tem-se:

$$U = U^* = \int_V \frac{\sigma^2}{2E} dV = \iiint \frac{T^2}{2EA^2} dA dx = \int_0^L \frac{T^2}{2EA} dx = \frac{T^2 L}{2EA} \tag{II.57}$$

Igualando U com W, tem-se:

$$U = W \Leftrightarrow \frac{T^2}{2k} = \frac{T^2 L}{2EA} \Leftrightarrow k = \frac{EA}{L} \tag{II.58}$$

A equação (II.58) mostra o valor da constante (k) de rigidez de uma barra de treliça, função de sua geometria e de seu material.

- ***Cálculo dos esforços restauradores***

O único tipo de esforço envolvido neste elemento é o de tração ou compressão. Desta forma, para calcular os esforços restauradores, a informação do comprimento da barra na configuração inicial (L) e deformada (L') são suficientes.

Pode-se calcular a tração acumulada (T') a partir dos resultados das equações (II.53) e (II.58), da seguinte forma:

$$T' = k(L' - L) \tag{II.59}$$

No caso de haver pré-tração, é necessário contabilizar seu efeito antes de o carregamento ser aplicado. Modificando a equação (II.59) e denotando por T_o a pré-tração, o esforço restaurador pode ser calculado de maneira mais geral pela equação (II.60):

$$T' = T_o + k(L' - L) \tag{II.60}$$

- ***Matriz de Rigidez da Treliça***

Os coeficientes da matriz de rigidez derivam de um equacionamento de energia de deformação da treliça.

A equação de energia de deformação (*U*) para uma estrutura é dada pela expressão (II.61), obtida através da integração da expressão volumétrica da energia por unidade de volume para um comportamento elástico linear. De posse dessa equação é possível extrair cada termo da matriz de rigidez efetuando as substituições das deformações, calculadas em função dos deslocamentos e também admitindo que a treliça possua apenas rigidez axial.

$$U = \frac{E}{2}\int_{Volume} \varepsilon_{xx}^2 \cdot dV \tag{II.61}$$

Com base em [7], o valor ε_{xx} na treliça é dado pela equação (II.62)

$$\varepsilon_{xx} = \frac{\partial u}{\partial x} - \frac{\partial^2 v}{\partial x^2} y + \frac{1}{2}\left(\frac{\partial v}{\partial x}\right)^2 - \frac{\partial^2 w}{\partial x^2} z + \frac{1}{2}\left(\frac{\partial w}{\partial x}\right)^2 \tag{II.62}$$

Substituindo(II.61) em (II.62), obtém-se a equação (II.63):

$$U = \frac{EA}{2}\int_0^L \left(\frac{\partial u}{\partial x}\right)^2 dx + \frac{EA}{2}\int_0^L \frac{\partial u}{\partial x}\left(\frac{\partial v}{\partial x}\right)^2 dx + \frac{EA}{2}\int_0^L \frac{\partial u}{\partial x}\left(\frac{\partial w}{\partial x}\right)^2 dx \tag{II.63}$$

Considerando a barra inicialmente alinhada com o eixo *x* do sistema de coordenadas, e os valores de deformações pequenos, pode-se escrever:

$$\sigma_{xx} = E\varepsilon_{xx} \Rightarrow \frac{T}{A} = E\frac{\partial u}{\partial x} \Rightarrow T = EA\frac{\partial u}{\partial x} \tag{II.64}$$

O resultado da equação (II.64) permite que se escreva a expressão da energia de deformação da forma mostrada na equação (II.65).

$$U = \frac{EA(\delta_4 - \delta_1)^2}{2}\frac{}{L} + \frac{T}{2}\frac{(\delta_5 - \delta_2)^2}{L} + \frac{T}{2}\frac{(\delta_6 - \delta_3)^2}{L} \tag{II.65}$$

Utilizando o Primeiro Teorema de Castigliano [8], que pode ser descrito pela equação (II.66), é possível obter a matriz de rigidez do elemento de treliça.

$$S_{ij} = \frac{\partial^2 U}{\partial q_i \partial q_j} \tag{II.66}$$

Organizam-se então os coeficientes de tal forma que os termos que possuem a tração T constituem a chamada matriz de rigidez geométrica ($\mathbf{K_G}$), e os termos restantes fazem parte da matriz de rigidez constitutiva elástica, dentro das hipóteses adotadas ($\mathbf{K_E}$). Sendo assim, a matriz de rigidez do elemento - equação (II.67) - é dada pela soma das matrizes (II.68) e (II.69).

$$\mathbf{K} = \mathbf{K_E} + \mathbf{K_G} \tag{II.67}$$

É importante notar que as matrizes aqui apresentadas estão escritas com base no sistema de coordenadas local do elemento e, portanto, é necessário realizar uma rotação antes de inseri-las na matriz de rigidez global. (Apêndice A)

$$\mathbf{K_E} = \begin{bmatrix} \frac{EA}{L} & 0 & 0 & -\frac{EA}{L} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{EA}{L} & 0 & 0 & \frac{EA}{L} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{II.68}$$

$$\mathbf{K}_G = \frac{T}{L}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 \end{bmatrix} \tag{II.69}$$

II.3.2 Elemento tipo Cabo – "Cable"

O elemento do tipo cabo, como esquematizado na Figura 8, possui seis graus de liberdade, sendo todos eles de translação. Graus de liberdade de rotação não são contemplados, e, portanto não necessariamente existe continuidade de ângulo de inclinação entre esses elementos. A continuidade ângulo de inclinação é garantida somente no interior de cada elemento, por sua função de forma.

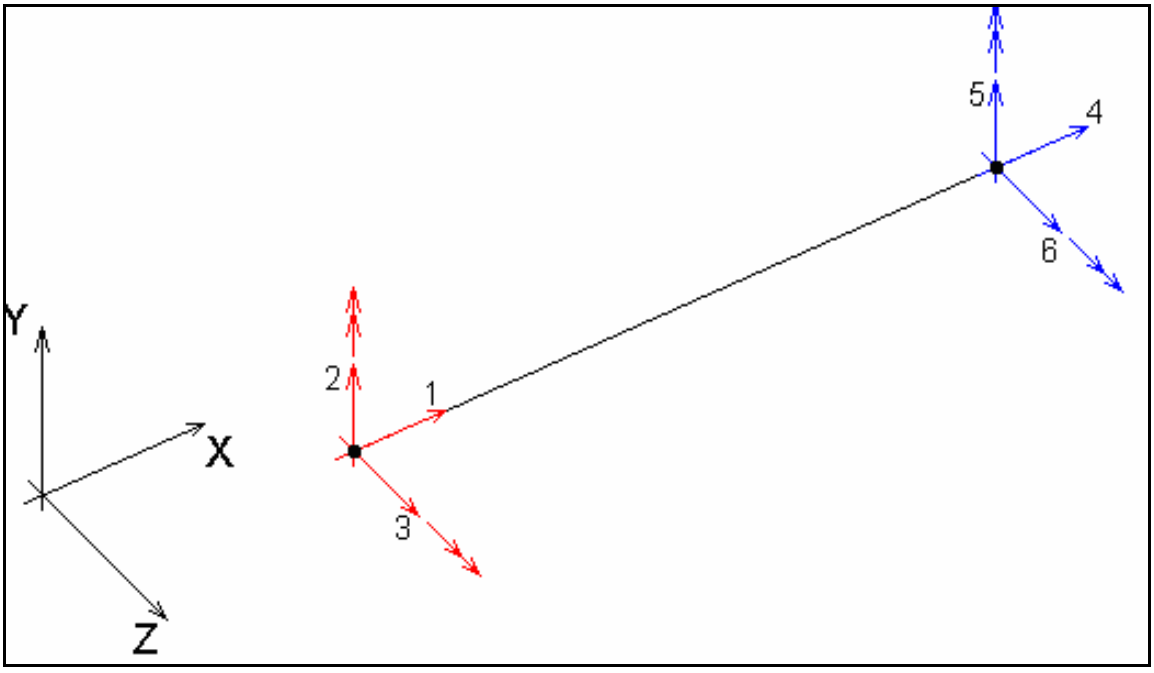


Figura 8 – Elemento do Tipo Cabo com 10 graus de liberdade

Os deslocamentos para cada ponto dessa estrutura estão parametrizados em função do parâmetro *t*, e são dados pelas equações (II.70), (II.71) e (II.72) , onde δ_i e φ_i indicam o deslocamento e a função de forma para o grau de liberdade *i*, respectivamente. Assim escreve-se as coordenadas *x, y* e *z* do elemento em função de *t*.

$$x(t) = \varphi_1(t)\delta_1 + \varphi_4(t)\delta_4 + t \tag{II.70}$$

$$y(t) = \varphi_2(t)\delta_2 + \varphi_5(t)\delta_5 \tag{II.71}$$

$$w(t) = \varphi_3(t)\delta_3 + \varphi_6(t)\delta_6 \tag{II.72}$$

Assim, pode-se escrever a curva parametrizada - equação (II.73) - desse elemento em função do parâmetro *t*, que pode variar entre 0 e *L* (onde L é comprimento inicial do elemento).

$$\Gamma(t) = \big(u(t), v(t), w(t)\big) \tag{II.73}$$

- ***Determinação das Funções de Forma***

As funções de forma têm o aspecto mostrado na equação (II.74), onde a, b e c são variáveis a serem determinadas a partir das condições de contorno para cada grau de liberdade.

$$\varphi(t) = at^2 + bt + c \tag{II.74}$$

A determinação das constantes *a*, *b* e *c* é feita impondo-se um deslocamento unitário na direção do grau de liberdade em questão e posteriormente, através das condições de contorno impostas por este deslocamento, determinam-se as constantes e consequentemente a função de forma para o respectivo grau de liberdade, como pode ser visto na Tabela 3.

Tabela 3 – Funções de Forma para o elemento de Cabo

Grau de liberdade 1	**Grau de liberdade 2**	**Grau de liberdade 3**
$\varphi_1(0)=1 \quad \varphi_1(L)=0$	$\varphi_2(0)=1 \quad \varphi_2'(0)=0 \quad \varphi_2(L)=0$	$\varphi_3(0)=1 \quad \varphi_3'(0)=0 \quad \varphi_3(L)=0$
$\varphi_1(t)=1-\frac{t}{L}$	$\varphi_2(t)=1-\frac{t^2}{L^2}$	$\varphi_3(t)=1-\frac{t^2}{L^2}$
Grau de liberdade 4	**Grau de liberdade 5**	**Grau de liberdade 6**
$\varphi_4(0)=0 \quad \varphi_4(L)=1$	$\varphi_5(0)=0 \quad \varphi_5(L)=1 \quad \varphi_5'(L)=0$	$\varphi_6(0)=0 \quad \varphi_6(L)=1 \quad \varphi_6'(L)=0$
$\varphi_4(t)=\frac{t}{L}$	$\varphi_5(t)=\frac{2t}{L}-\frac{t^2}{L^2}$	$\varphi_6(t)=\frac{2t}{L}-\frac{t^2}{L^2}$

Substituindo nas equações (II.70), (II.71) e (II.72) as funções de forma já calculadas, pode-se escrever as equações (II.75), (II.76) e (II.77).

$$u(t)=\left(1-\frac{t}{L}\right)\cdot\delta_1+\frac{t}{L}\cdot\delta_4+t \tag{II.75}$$

$$v(t)=\left(1-\frac{t^2}{L^2}\right)\cdot\delta_2+\left(\frac{2t}{L}-\frac{t^2}{L^2}\right)\cdot\delta_5 \tag{II.76}$$

$$w(t)=\left(1-\frac{t^2}{L^2}\right)\cdot\delta_3+\left(\frac{2t}{L}-\frac{t^2}{L^2}\right)\cdot\delta_6 \tag{II.77}$$

- ***Cálculo do comprimento do cabo na configuração deformada***

O novo comprimento na configuração deformada pode ser calculado através da equação:

$$L'=\int_0^L\left|\frac{d\Gamma}{dt}\right|dt \tag{II.78}$$

Desenvolvendo o integrando, obtém-se:

$$\left|\frac{d\Gamma}{dx}\right| = \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2 + \left(\frac{dz}{dt}\right)^2} = \\ = \sqrt{\left(\frac{\delta_4 - \delta_1}{L} + 1\right)^2 + \left[-\frac{2t}{L^2}\delta_2 + \left(\frac{2}{L} - \frac{2t}{L^2}\right)\delta_5\right]^2 + \left[-\frac{2t}{L^2}\delta_3 + \left(\frac{2}{L} - \frac{2t}{L^2}\right)\delta_6\right]^2} \tag{II.79}$$

A integral é muito mais complicada do que no caso do elemento de treliça, e, portanto recorreu-se a um método numérico para seu cálculo. O método utilizado foi o de Romberg. (Apêndices B e C).

- ***Cálculo da constante elástica para a direção do cabo***

O cálculo da constante elástica *k* para a direção do elemento de cabo é semelhante ao de treliça, e seu valor é o mesmo desenvolvido na equação (II.58).

- ***Cálculo dos esforços restauradores***

Analogamente ao elemento de treliça, o único tipo de esforço envolvido nesse elemento são esforços axiais. Assim, o cálculo dos esforços restauradores é feito de maneira semelhante.Como é feito na equação (II.60).

A aplicação dos esforços restauradores, nesse caso, não ocorreria exatamente na direção que une os nós do elemento, mas sim na direção tangente à curva parametrizada do cabo. No entanto, admite-se que essa diferença de direção é desprezível para um bom refinamento de

malha, e por isso, a aplicação dos esforços foi feita na própria direção que une os nós do elemento.

- ***Matriz de Rigidez do Cabo***

A matriz de rigidez do cabo, como a da treliça, pode ser dividida em duas partes: uma constitutiva e outra geométrica.

Utilizando as equações (II.61) e (II.62), em um procedimento análogo ao aplicado no elemento de treliça, pode-se obter a expressão (II.80), para a energia de deformação no elemento de cabo, em função dos deslocamentos em cada grau de liberdade:

$$U=\frac{EA\left(\delta_4-\delta_1\right)^2}{2}\frac{}{L}+\frac{T}{2}\left\{\frac{4}{3L}\delta_2^2-\frac{4}{3L}\delta_2\delta_5+\frac{4}{3L}\delta_5^2+\frac{4}{3L}\delta_3^2-\frac{4}{3L}\delta_3\delta_6+\frac{4}{3L}\delta_6^2\right\} \qquad \text{(II.80)}$$

Utilizando o Primeiro Teorema de Castigliano, descrito pela equação (II.66), obtém-se a matriz de rigidez constitutiva elástica - equação (II.81) - e geométrica - equação (II.82) - para o elemento de cabo. A matriz de rigidez total para o elemento de cabo será a soma da parte geométrica com a parte constitutiva elástica. Essas matrizes estão escritas no sistema de coordenadas local do elemento. A aplicação de uma rotação será necessária para montar a matriz de rigidez global (Apêndice A).

$$\mathbf{K_E} = \begin{bmatrix} \frac{EA}{L} & 0 & 0 & -\frac{EA}{L} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{EA}{L} & 0 & 0 & \frac{EA}{L} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{II.81}$$

$$\mathbf{K_G} = \frac{T}{L}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{4}{3} & 0 & 0 & -\frac{2}{3} & 0 \\ 0 & 0 & \frac{4}{3} & 0 & 0 & -\frac{2}{3} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{2}{3} & 0 & 0 & \frac{4}{3} & 0 \\ 0 & 0 & -\frac{2}{3} & 0 & 0 & \frac{4}{3} \end{bmatrix} \tag{II.82}$$

II.3.3 Elemento tipo Pórtico – "Beam"

O elemento do tipo pórtico, como esquematizado na Figura 9 possui doze graus de liberdade, sendo seis deles de rotação e seis de translação. Devido à existência dos graus de liberdade de rotação, é garantida a continuidade angular entre esses elementos, isto é, o ângulo entre dois elementos consecutivos da malha inicial é mantido, mesmo que haja grandes deslocamentos da estrutura como um todo. As funções de forma garantem, também, a continuidade angular no interior de cada elemento.

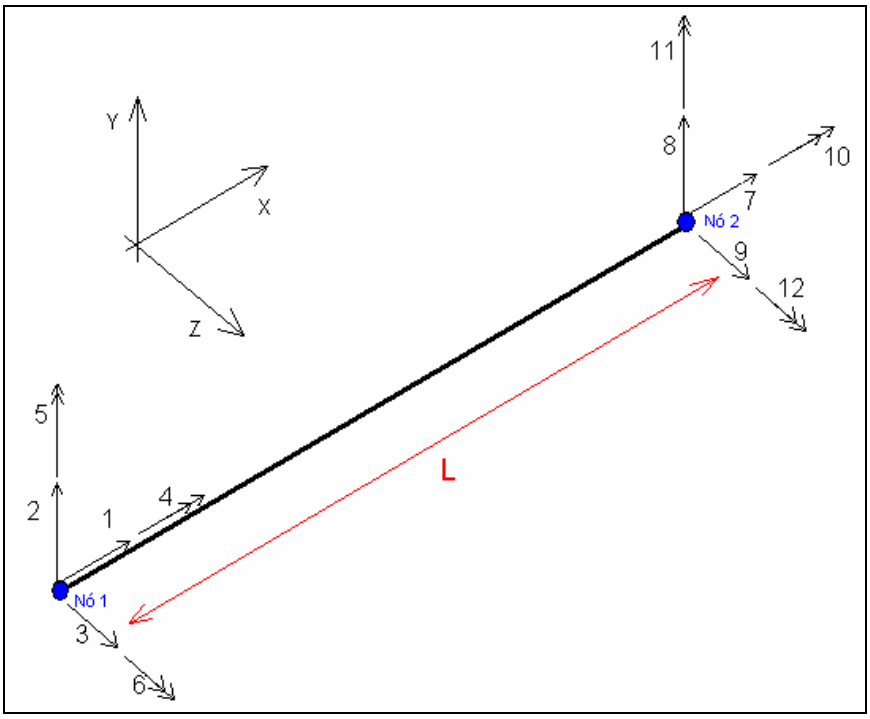


Figura 9 – Elemento do Tipo Pórtico com 12 graus de liberdade

O elemento desenvolvido possui algumas hipóteses simplificadoras em sua formulação, descritas abaixo:

- Não foi considerado o acoplamento entre flexão e torção.
- A flexão é independente e para cada plano de flexão, e não foi considerado acoplamento entre eles.
- Em caso de torção, não ocorre empenamento da seção transversal.
- Na flexão, as seções planas permanecem planas.
- As rotações no espaço são de pequeno valor, de modo que são consideradas comutativas, o que não ocorreria para o caso de grandes rotações no espaço.

As posições para cada ponto dessa estrutura estão parametrizados em função de *t*, que deve variar de 0 a L, a fim de obter toda a linha elástica, e são dados pelas equações (II.83), (II.84) e (II.85), onde δ_i e φ_i indicam o deslocamento e a função de forma para o grau de liberdade *i*,

respectivamente. As funções *x(t)*, *y(t)* e *z(t)* indicam as coordenadas em função do parâmetro *t*.

$$x(t) = \varphi_1(t)\delta_1 + \varphi_7(t)\delta_7 + t \tag{II.83}$$

$$y(t) = \varphi_2(t)\delta_2 + \varphi_6(t)\delta_6 + \varphi_8(t)\delta_8 + \varphi_{12}(t)\delta_{12} \tag{II.84}$$

$$z(t) = \varphi_3(t)\delta_3 + \varphi_5(t)\delta_5 + \varphi_9(t)\delta_9 + \varphi_{11}(t)\delta_{11} \tag{II.85}$$

Assim, pode-se escrever a curva parametrizada - equação (II.86) - desse elemento em função do parâmetro x, que pode variar entre 0 e *L* (comprimento inicial do elemento).

$$\Gamma(t) = \left[x(t), y(t), z(t)\right] \tag{II.86}$$

- ***Determinação das Funções de Forma***

As funções de forma possuem diferentes graus polinomiais, dependendo do grau de liberdade que retratam.Para os graus de liberdade de deslocamento axial e torcional, são utilizadas funções de primeiro grau – equação (II.87), enquanto que para os graus de liberdade de translação da direção transversal ao pórtico, e rotação causando momento fletor, são utilizadas funções de terceiro grau – equação (II.88)

$$\varphi(t) = at + b \tag{II.87}$$

$$\varphi(t) = ct^3 + dt^2 + et + f \tag{II.88}$$

A determinação das constantes *a*, *b, c, d, e* e *f* é feita impondo-se um deslocamento unitário na direção do grau de liberdade em questão e posteriormente, através das condições de contorno impostas por este deslocamento, determinam-se as constantes e consequentemente a função

de forma para o respectivo grau de liberdade. A Tabela 4 mostra as condições de contorno e funções de forma obtidas para cada grau de liberdade do elemento.

Tabela 4 – Funções de Forma para o elemento de Pórtico

Grau de liberdade 1	**Grau de liberdade 2**	**Grau de liberdade 3**
$\varphi_1(0)=1 \quad \varphi_1(L)=0$ $\varphi_1(t)=1-\frac{t}{L}$	$\varphi_2(0)=1 \quad \varphi_2(L)=0$ $\left.\frac{d\varphi_2}{dt}\right\|_0=0 \quad \left.\frac{d\varphi_2}{dt}\right\|_L=0$ $\varphi_2(t)=2\left(\frac{t}{L}\right)^3-3\left(\frac{t}{L}\right)^2+1$	$\varphi_3(0)=1 \quad \varphi_3(L)=0$ $\left.\frac{d\varphi_3}{dt}\right\|_0=0 \quad \left.\frac{d\varphi_3}{dt}\right\|_L=0$ $\varphi_3(t)=2\left(\frac{t}{L}\right)^3-3\left(\frac{t}{L}\right)^2+1$
Grau de liberdade 4	**Grau de liberdade 5**	**Grau de liberdade 6**
$\varphi_4(0)=1 \quad \varphi_4(L)=0$ $\varphi_4(t)=1-\frac{t}{L}$	$\varphi_5(0)=0 \quad \varphi_5(L)=0$ $\left.\frac{d\varphi_5}{dt}\right\|_0=-1 \quad \left.\frac{d\varphi_5}{dt}\right\|_L=0$ $\varphi_5(t)=-\frac{t^3}{L^2}+2\frac{t^2}{L}-t$	$\varphi_6(0)=0 \quad \varphi_6(L)=0$ $\left.\frac{d\varphi_6}{dt}\right\|_0=1 \quad \left.\frac{d\varphi_6}{dt}\right\|_L=0$ $\varphi_6(t)=\frac{t^3}{L^2}-2\frac{t^2}{L}+t$
Grau de liberdade 7	**Grau de liberdade 8**	**Grau de liberdade 9**
$\varphi_7(0)=0 \quad \varphi_7(L)=1$ $\varphi_7(t)=\frac{t}{L}$	$\varphi_8(0)=0 \quad \varphi_8(L)=1$ $\left.\frac{d\varphi_8}{dt}\right\|_0=0 \quad \left.\frac{d\varphi_8}{dt}\right\|_L=0$ $\varphi_8(t)=-2\left(\frac{t}{L}\right)^3+3\left(\frac{t}{L}\right)^2$	$\varphi_9(0)=0 \quad \varphi_9(L)=1$ $\left.\frac{d\varphi_9}{dt}\right\|_0=0 \quad \left.\frac{d\varphi_9}{dt}\right\|_L=0$ $\varphi_9(t)=-2\left(\frac{t}{L}\right)^3+3\left(\frac{t}{L}\right)^2$
Grau de liberdade 10	**Grau de liberdade 11**	**Grau de liberdade 12**
$\varphi_{10}(0)=0 \quad \varphi_{10}(L)=1$ $\varphi_{10}(t)=\frac{t}{L}$	$\varphi_{11}(0)=0 \quad \varphi_{11}(L)=0$ $\left.\frac{d\varphi_{11}}{dt}\right\|_0=0 \quad \left.\frac{d\varphi_{11}}{dt}\right\|_L=1$ $\varphi_{11}(t)=-\frac{t^3}{L^2}+\frac{t^2}{L}$	$\varphi_{12}(0)=0 \quad \varphi_{12}(L)=0$ $\left.\frac{d\varphi_{12}}{dt}\right\|_0=0 \quad \left.\frac{d\varphi_{12}}{dt}\right\|_L=-1$ $\varphi_{12}(t)=\frac{t^3}{L^2}-\frac{t^2}{L}$

Substituindo nas equações (II.83), (II.84) e (II.89) as funções de forma já calculadas, pode-se escrever as equações (II.90), (II.91) e (II.92).

$$x(t)=\left(1-\frac{t}{L}\right)\delta_1+\frac{t}{L}\delta_7+t \tag{II.90}$$

$$y(t)=\left[2\left(\frac{t}{L}\right)^3-3\left(\frac{t}{L}\right)^2+1\right]\delta_2+\left[\frac{t^3}{L^2}-2\frac{t^2}{L}+t\right]\delta_6+\left[3\left(\frac{t}{L}\right)^2-2\left(\frac{t}{L}\right)^3\right]\delta_8+\left[\frac{t^3}{L^2}-\frac{t^2}{L}\right]\delta_{12} \tag{II.91}$$

$$z(t)=\left[2\left(\frac{t}{L}\right)^3-3\left(\frac{t}{L}\right)^2+1\right]\delta_3+\left[2\frac{t^2}{L}-\frac{t^3}{L^2}-t\right]\delta_5+\left[3\left(\frac{t}{L}\right)^2-2\left(\frac{t}{L}\right)^3\right]\delta_9+\left[\frac{t^2}{L}-\frac{t^3}{L^2}\right]\delta_{11} \tag{II.92}$$

- ***Cálculo do comprimento do pórtico na configuração deformada***

O novo comprimento na configuração deformada pode ser calculado através da equação:

$$L'=\int_0^L\left|\frac{d\Gamma}{dt}\right|dt \tag{II.93}$$

Desenvolvendo o integrando, obtém-se:

$$\left|\frac{d\Gamma}{dt}\right|=\sqrt{\left(\frac{dx(t)}{dt}\right)^2+\left(\frac{dy(t)}{dt}\right)^2+\left(\frac{dz(t)}{dt}\right)^2} \tag{II.94}$$

Para calcular a função que será o integrando (equação (II.93)), é necessário calcular as derivadas de cada um dos componentes *x*, *y* e *z*, em relação ao parâmetro *t*.

$$\frac{dx(t)}{dt}=\frac{(\delta_7-\delta_1)}{L}+1 \tag{II.95}$$

$$\frac{dy(t)}{dt}=\frac{d\varphi_2(t)}{dt}\delta_2+\frac{d\varphi_6(t)}{dt}\delta_6+\frac{d\varphi_8(t)}{dt}\delta_8+\frac{d\varphi_{12}(t)}{dt}\delta_{12} \tag{II.96}$$

$$\frac{dz(t)}{dt}=\frac{d\varphi_3(t)}{dt}\delta_3+\frac{d\varphi_5(t)}{dt}\delta_5+\frac{d\varphi_9(t)}{dt}\delta_9+\frac{d\varphi_{11}(t)}{dt}\delta_{11} \tag{II.97}$$

Calculando a derivada de cada uma das funções de forma, é possível obter as seguintes expressões:

$$\frac{d\varphi_2(t)}{dt} = \frac{d\varphi_3(t)}{dt} = 6\frac{t^2}{L^3} - 6\frac{t}{L^2} \tag{II.98}$$

$$\frac{d\varphi_8(t)}{dt} = \frac{d\varphi_9(t)}{dt} = -6\frac{t^2}{L^3} + 6\frac{t}{L^2} \tag{II.99}$$

$$\frac{d\varphi_5(t)}{dt} = -\frac{d\varphi_6(t)}{dt} = -3\frac{t^2}{L^2} + 4\frac{t}{L} - 1 \tag{II.100}$$

$$\frac{d\varphi_{11}(t)}{dt} = -\frac{d\varphi_{12}(t)}{dt} = -3\frac{t^2}{L^2} + 2\frac{t}{L} \tag{II.101}$$

Substituindo as equações (II.98) a (II.101) nas equações (II.95) a (II.97), e essas na equação (II.94), pode-se obter uma expressão para o integrando da equação (II.93), para o cálculo do comprimento na configuração deformada do pórtico.

- ***Cálculo dos esforços restauradores***

Nesse elemento podem estar presentes quatro tipos de esforços: tração (ou compressão), força cortante, momento fletor e momento torçor. Cada um dos três esforços pode ser calculado a partir da equação da linha elástica do pórtico. Para o esforço normal, o cálculo é feito de maneira semelhante ao elemento de treliça.

Pode-se calcular a tração acumulada (*T'*) a partir dos resultados da equação (II.93) da seguinte forma:

$$T' = k(L' - L) \tag{II.102}$$

Na equação (II.102) *k* representa um valor de rigidez axial equivalente para o pórtico, que será desenvolvido na próxima seção.

No caso de haver pré-tração, é necessário contabilizar seu efeito antes de o carregamento ser aplicado. Modificando a equação (II.102) e denotando por T_o a pré-tração, o esforço restaurador normal pode ser calculado de maneira mais geral pela equação (II.103):

$$T' = T_o + k(L' - L) \tag{II.103}$$

Para o cálculo dos momentos fletores acumulados no pórtico, e que portanto deverão ser aplicados como esforços restauradores da estrutura, é necessário utilizar a relação entre momento fletor e curvatura, encontrada em Gere [8].

$$M = EI\kappa \tag{II.104}$$

Onde: M é o momento fletor em uma seção transversal do pórtico;

EI é a rigidez flexional do pórtico em uma seção transversal;

κ é a curvatura do pórtico em uma seção transversal.

Uma vez obtidos os componentes *y(t)* e *z(t)* da equação da linha elástica parametrizada, é possível calcular individualmente a curvatura que ocorre em cada um desses planos individualmente e, após isso, através de uma soma de vetores, realizar o cálculo da curvatura total no espaço tridimensional. A equação (II.105) mostra como calcular a curvatura geometricamente exata em um plano a partir da função da linha elástica e, portanto, pode ser aplicada individualmente para *y(t)* e *z(t)*. A demonstração da equação (II.105) se encontra no (Apêndice D).

$$\kappa = \left(\frac{1}{R}\right) = \frac{\dfrac{d^2 f(t)}{dx^2}}{\left[1 + \left(\dfrac{df(t)}{dx}\right)^2\right]^{3/2}} \tag{II.105}$$

Onde: R é o raio de curvatura do pórtico na coordenada x;

$f(t)$ é a função da linha elástica do pórtico;

Assim, é possível realizar o cálculo dos momentos fletores restauradores, utilizando-se as equações (II.104) e (II.105), para $y(t)$ e $z(t)$ individualmente e, portanto, calculando momentos fletores que atuam individualmente em dois planos distintos. O cálculo da segunda derivada da função da linha elástica pode ser feito de maneira análoga ao cálculo da primeira derivada (equações (II.90), (II.91) e (II.106)), porém desta vez apenas para as funções $y(t)$ e $z(t)$. Assim:

$$\frac{d^2 y(t)}{dt^2} = \frac{d^2 \varphi_2(t)}{dt^2}\delta_2 + \frac{d^2 \varphi_6(t)}{dt^2}\delta_6 + \frac{d^2 \varphi_8(t)}{dt^2}\delta_8 + \frac{d^2 \varphi_{12}(t)}{dt^2}\delta_{12} \tag{II.107}$$

$$\frac{d^2 z(t)}{dt^2} = \frac{d^2 \varphi_3(t)}{dt^2}\delta_3 + \frac{d^2 \varphi_5(t)}{dt^2}\delta_5 + \frac{d^2 \varphi_9(t)}{dt^2}\delta_9 + \frac{d^2 \varphi_{11}(t)}{dt^2}\delta_{11} \tag{II.108}$$

Para o cálculo dos esforços cortantes de maneira exata, é possível derivar a equação (II.105), pois é possível mostrar que:

$$\frac{dM(t)}{dt} = V(t) \tag{II.109}$$

Onde: $M(t)$ é o momento fletor em uma seção de coordenada x;

$V(t)$ é a força cortante nessa mesma coordenada x.

Assim, deduz-se que:

$$V(t) = EI \frac{1}{\left[1+\left(\frac{df(t)}{dt}\right)^2\right]^{5/2}} \left\{ -\frac{d^3 f(t)}{dt^3}\left[1+\left(\frac{df(t)}{dt}\right)^2\right] + 3\left(\frac{d^2 f(t)}{dt^2}\right)^2 \frac{df(t)}{dt} \right\} \tag{II.110}$$

O cálculo da segunda e terceira derivada da função da linha elástica pode ser feito de maneira análoga ao cálculo da primeira derivada (equações (II.90), (II.91) e (II.111)), porém desta vez apenas para as funções *y(t)* e *z(t)*, necessárias para o cálculo de *M(t)* e *V(t)*. Assim:

$$\frac{d^2 v(t)}{dt^2} = \frac{d^2 \varphi_2(t)}{dt^2}\delta_2 + \frac{d^2 \varphi_6(t)}{dt^2}\delta_6 + \frac{d^2 \varphi_8(t)}{dt^2}\delta_8 + \frac{d^2 \varphi_{12}(t)}{dt^2}\delta_{12} \tag{II.112}$$

$$\frac{d^2 w(t)}{dt^2} = \frac{d^2 \varphi_3(t)}{dt^2}\delta_3 + \frac{d^2 \varphi_5(t)}{dt^2}\delta_5 + \frac{d^2 \varphi_9(t)}{dt^2}\delta_9 + \frac{d^2 \varphi_{11}(t)}{dt^2}\delta_{11} \tag{II.113}$$

As derivadas segundas em relação a *x* das funções de forma estão descritas a seguir.

$$\frac{d^2 \varphi_2(t)}{dt^2} = \frac{d^2 \varphi_3(t)}{dt^2} = -\frac{d^2 \varphi_8(t)}{dt^2} = -\frac{d^2 \varphi_9(t)}{dt^2} = \frac{12}{L^3}t - \frac{6}{L^2} \tag{II.114}$$

$$\frac{d^2 \varphi_5(t)}{dt^2} = -\frac{d^2 \varphi_6(t)}{dt^2} = -\frac{6}{L^2}t + \frac{4}{L} \tag{II.115}$$

$$\frac{d^2 \varphi_{11}(t)}{dt^2} = -\frac{d^2 \varphi_{12}(t)}{dt^2} = -\frac{6}{L^2}t + \frac{2}{L} \tag{II.116}$$

Para a derivada terceira:

$$\frac{d^3 y(t)}{dt^3} = \frac{d^3 \varphi_2(t)}{dt^3}\delta_2 + \frac{d^3 \varphi_6(t)}{dt^3}\delta_6 + \frac{d^3 \varphi_8(t)}{dt^3}\delta_8 + \frac{d^3 \varphi_{12}(t)}{dt^3}\delta_{12} \tag{II.117}$$

$$\frac{d^3 z(t)}{dt^3} = \frac{d^3 \varphi_3(t)}{dt^3}\delta_3 + \frac{d^3 \varphi_5(t)}{dt^3}\delta_5 + \frac{d^3 \varphi_9(t)}{dt^3}\delta_9 + \frac{d^3 \varphi_{11}(t)}{dt^3}\delta_{11} \tag{II.118}$$

As derivadas terceiras em relação a *x* das funções de forma estão descritas a seguir.

$$\frac{d^3 \varphi_2(t)}{dt^3} = \frac{d^3 \varphi_3(t)}{dt^3} = -\frac{d^3 \varphi_8(t)}{dt^3} = -\frac{d^3 \varphi_9(t)}{dt^3} = \frac{12}{L^3} \tag{II.119}$$

$$\frac{d^3\varphi_5(t)}{dt^3} = -\frac{d^3\varphi_6(t)}{dt^3} = \frac{d^3\varphi_{11}(t)}{dt^3} = -\frac{d^3\varphi_{12}(t)}{dt^3} = -\frac{6}{L^2} \tag{II.120}$$

Para o cálculo dos momentos torçores restauradores, utiliza-se a seguinte relação:

$$M_T(t) = \frac{GJ}{L}\theta(t) \tag{II.121}$$

Onde: $M_T(t)$ é o momento torçor em uma coordenada *x* do pórtico;

GJ/L é a rigidez torcional equivalente do pórtico;

θ(t) é o ângulo de torção.

Assim, o momento torçor restaurador, ou seja, acumulado no pórtico, pode ser calculado se o ângulo de torção for conhecido, que é o que ocorre, pois os esforços restauradores são calculados após cada iteração e, portanto, os deslocamentos em cada grau de liberdade são conhecidos. Isso também permite o cálculo das funções *y(t)* e *z(t)* para os cálculos de *M(t)* e *V(t)*.

Inserção dos esforços restauradores

Para inserir os esforços restauradores (esforços internos acumulados no elemento) no vetor de esforços global do sistema são feitas algumas simplificações com relação à direção da inserção desses esforços. Apesar de os elementos sofrerem rotações, a direção tida como normal é considerada a direção da reta que une os dois nós. A direção cortante, por sua vez, é considerada como sendo ortogonal a essa aproximação de direção normal.

A Figura 10 mostra em azul as direções exatas e, em vermelho, as direções aproximadas para as direções normal e cortante em um plano da linha elástica. As mesmas direções

aproximadas definem as direções de atuação dos momentos fletores e torçor restauradores para cada nó do elemento.

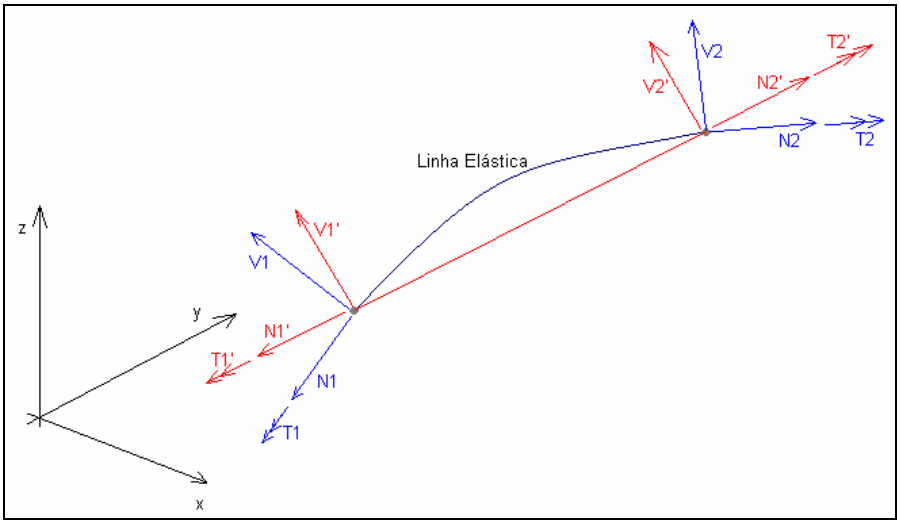


Figura 10 – Aproximação nas direções dos esforços restauradores

Note que com o refinamento da malha esse erro torna-se cada vez menor para cada elemento e, portanto, não se torna um problema na modelagem dos problemas físicos.

- ***Matriz de Rigidez do Pórtico***

Os coeficientes da matriz de rigidez derivam de um equacionamento de energia de deformação do pórtico. A expressão (II.109) descreve a energia de deformação a partir dos valores de deformação longitudinal, e distorções no plano de cada seção transversal de um pórtico.

$$U = \frac{E}{2}\int_V \varepsilon_{xx}^2 dV + \frac{G}{2}\int_V (\varepsilon_{xy}^2 + \varepsilon_{xz}^2)\, dV \qquad \text{(II.122)}$$

É necessário escrever uma expressão para calcular as deformações ε_{xx}, ε_{xy} e ε_{xz} a partir dos deslocamentos.

É possível demonstrar, a partir de parâmetros geométricos, a equação (II.123) que representa o valor da deformação em um ponto da estrutura, devido à superposição dos seguintes efeitos: flexão que ocorre no plano cuja normal é *y* e no plano cuja normal é *z* (sistemas de coordenadas locais) e devido ao deslocamento axial das seções transversais quando sujeitas a esforços de tração ou compressão.

$$\varepsilon_{xx} = \frac{\partial u}{\partial x} - y\frac{\partial^2 v}{\partial x^2} + \frac{1}{2}\left(\frac{\partial v}{\partial x}\right)^2 + z\frac{\partial^2 w}{\partial x^2} + \frac{1}{2}\left(\frac{\partial w}{\partial x}\right)^2 \tag{II.123}$$

Onde: *u* é o deslocamento de um ponto da estrutura na direção *x*;

v é o deslocamento de um ponto da estrutura na direção *y*;

w é o deslocamento de um ponto da estrutura na direção *z*.

As expressões para o cálculo de *u*, *v* e *w* estão descritas a seguir.

$$u(t) = \varphi_1(t)\delta_1 + \varphi_7(t)\delta_7 \tag{II.124}$$

$$v(t) = \varphi_2(t)\delta_2 + \varphi_6(t)\delta_6 + \varphi_8(t)\delta_8 + \varphi_{12}(t)\delta_{12} \tag{II.125}$$

$$w(t) = \varphi_3(t)\delta_3 + \varphi_5(t)\delta_5 + \varphi_9(t)\delta_9 + \varphi_{11}(t)\delta_{11} \tag{II.126}$$

É possível escrever, também, relações puramente geométricas que relacionem a distorção com a rotação – equações (II.124) e (II.125).

$$\varepsilon_{xy}^2 = z^2\left(\frac{\partial \theta}{\partial x}\right)^2 \tag{II.127}$$

$$\varepsilon_{xz}^2 = y^2\left(\frac{\partial \theta}{\partial x}\right)^2 \tag{II.128}$$

Onde: θ é o ângulo de rotação da seção transversal.

A expressão para o cálculo de θ(t) está descrita a seguir.

$$\theta(t) = \varphi_4(t)\delta_4 + \varphi_{10}(t)\delta_{10} \tag{II.129}$$

A equação (II.122) pode ser escrita da forma:

$$U = \frac{E}{2}\int\int_A \varepsilon_{xx}^2 dAdx + \frac{G}{2}\int\int_A (\varepsilon_{xy}^2 + \varepsilon_{xz}^2) dAdx \tag{II.130}$$

Onde: E é o módulo de elasticidade do material do pórtico;

G é o módulo de cisalhamento do material do pórtico.

Substituindo as equações (II.123), (II.127) e (II.128) na equação (II.130), desprezando-se os termos de quarta ordem e considerando as integrais $\int_A ydA$, $\int_A zdA$ e $\int_A yzdA$ nulas, pela simetria dos típicos problemas que irão ser tratados, nos quais os momentos de primeira ordem e produtos de inércia são nulos, é possível obter:

$$\begin{aligned} U = {} & \frac{EA}{2}\int_0^L \left(\frac{\partial u}{\partial x}\right)^2 dx + \frac{EA}{2}\int_0^L \frac{\partial u}{\partial x}\left(\frac{\partial v}{\partial x}\right)^2 dx + \frac{EA}{2}\int_0^L \frac{\partial u}{\partial x}\left(\frac{\partial w}{\partial x}\right)^2 dx \\ & + \frac{EI}{2}\int_0^L \left(\frac{\partial^2 v}{\partial x^2}\right)^2 dx + \frac{EI}{2}\int_0^L \left(\frac{\partial^2 w}{\partial x^2}\right)^2 dx + \frac{GJ}{2}\int_0^L \left(\frac{\partial \theta}{\partial x}\right)^2 dx \end{aligned} \tag{II.131}$$

As derivadas da equação acima podem ser calculadas a partir das equações polinomiais (II.124), (II.125), (II.132) e (II.129). Substituindo-as na equação (II.133) obtém-se uma expressão para a energia interna (U) de deformação do pórtico. Aplicando-se o Primeiro Teorema de Castigliano, para obtenção dos esforços devido a deslocamentos unitários em diversos graus de liberdade, é possível obter os coeficientes de influência de rigidez entre cada par de graus de liberdade. Podem-se escrever esses coeficientes na forma matricial (matriz de rigidez) e, dividindo essa matriz em duas partes, sendo uma delas formada por

termos que contenham explicitamente a tração *T* acumulada no elemento e, outra, contendo os outros termos, são identificadas respectivamente a matriz de rigidez geométrica e a matriz de rigidez constitutiva. Essas estão escritas a seguir, sendo K_G a matriz de rigidez geométrica, e K_C a matriz de rigidez constitutiva.

$$\mathbf{K_C} = \begin{bmatrix} \mathbf{K_1} & \mathbf{K}_2 \\ \mathbf{K}_3 & \mathbf{K}_4 \end{bmatrix} \tag{II.134}$$

Onde:

$$\mathbf{K}_1 = \begin{bmatrix} \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{12EI}{L^3} & 0 & 0 & 0 & \frac{6EI}{L^2} \\ 0 & 0 & \frac{12EI}{L^3} & 0 & -\frac{6EI}{L^2} & 0 \\ 0 & 0 & 0 & \frac{GJ}{L} & 0 & 0 \\ 0 & 0 & -\frac{6EI}{L^2} & 0 & \frac{4EI}{L} & 0 \\ 0 & \frac{6EI}{L^2} & 0 & 0 & 0 & \frac{4EI}{L} \end{bmatrix} \tag{II.135}$$

$$\mathbf{K}_2 = \begin{bmatrix} -\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{12EI}{L^3} & 0 & 0 & 0 & \frac{6EI}{L^2} \\ 0 & 0 & -\frac{12EI}{L^3} & 0 & -\frac{6EI}{L^2} & 0 \\ 0 & 0 & 0 & -\frac{GJ}{L} & 0 & 0 \\ 0 & 0 & \frac{6EI}{L^2} & 0 & \frac{2EI}{L} & 0 \\ 0 & -\frac{6EI}{L^2} & 0 & 0 & 0 & \frac{2EI}{L} \end{bmatrix} \tag{II.136}$$

$$\mathbf{K}_3 = \begin{bmatrix} -\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{12EI}{L^3} & 0 & 0 & 0 & -\frac{6EI}{L^2} \\ 0 & 0 & -\frac{12EI}{L^3} & 0 & \frac{6EI}{L^2} & 0 \\ 0 & 0 & 0 & -\frac{GJ}{L} & 0 & 0 \\ 0 & 0 & -\frac{6EI}{L^2} & 0 & \frac{2EI}{L} & 0 \\ 0 & \frac{6EI}{L^2} & 0 & 0 & 0 & \frac{2EI}{L} \end{bmatrix} \tag{II.137}$$

$$\mathbf{K}_4 = \begin{bmatrix} \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{12EI}{L^3} & 0 & 0 & 0 & -\frac{6EI}{L^2} \\ 0 & 0 & \frac{12EI}{L^3} & 0 & -\frac{6EI}{L^2} & 0 \\ 0 & 0 & 0 & \frac{GJ}{L} & 0 & 0 \\ 0 & 0 & -\frac{6EI}{L^2} & 0 & \frac{4EI}{L} & 0 \\ 0 & -\frac{6EI}{L^2} & 0 & 0 & 0 & \frac{4EI}{L} \end{bmatrix} \tag{II.138}$$

E:

$$\mathbf{K}_G = \begin{bmatrix} \mathbf{K_1} & \mathbf{K}_2 \\ \mathbf{K}_3 & \mathbf{K}_4 \end{bmatrix} \tag{II.139}$$

Com:

$$\mathbf{K}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{6T}{5L} & 0 & 0 & 0 & \frac{T}{10} \\ 0 & 0 & \frac{6T}{5L} & 0 & -\frac{T}{10} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{T}{10} & 0 & \frac{2TL}{15} & 0 \\ 0 & \frac{T}{10} & 0 & 0 & 0 & \frac{2TL}{15} \end{bmatrix} \tag{II.140}$$

$$\mathbf{K}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{6T}{5L} & 0 & 0 & 0 & \frac{T}{10} \\ 0 & 0 & -\frac{6T}{5L} & 0 & -\frac{T}{10} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{T}{10} & 0 & -\frac{TL}{30} & 0 \\ 0 & -\frac{T}{10} & 0 & 0 & 0 & -\frac{TL}{30} \end{bmatrix} \tag{II.141}$$

$$\mathbf{K}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{6T}{5L} & 0 & 0 & 0 & -\frac{T}{10} \\ 0 & 0 & -\frac{6T}{5L} & 0 & \frac{T}{10} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{T}{10} & 0 & -\frac{TL}{30} & 0 \\ 0 & \frac{T}{10} & 0 & 0 & 0 & -\frac{TL}{30} \end{bmatrix} \tag{II.142}$$

$$\mathbf{K}_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{6T}{5L} & 0 & 0 & 0 & -\frac{T}{10} \\ 0 & 0 & \frac{6T}{5L} & 0 & \frac{T}{10} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{T}{10} & 0 & \frac{2TL}{15} & 0 \\ 0 & -\frac{T}{10} & 0 & 0 & 0 & \frac{2TL}{15} \end{bmatrix} \quad \text{(II.143)}$$

II.4 Carregamentos – "Loads"

Para a análise estática de risers, devem ser contemplados o peso próprio da estrutura e efeitos de interação fluido-estrutura, provenientes do arrasto hidrodinâmico. Dessa forma, utilizando-se os dados usuais tidos como dados de entrada para um software de análise de risers, incluem-se na análise esses dois tipos de carregamentos.

Além disso, é possível inserir qualquer tipo de esforço nodal na malha gerada, dessa forma generalizando a estrutura do software para qualquer tipo de carregamento que venha a ser importante para análises específicas.

II.4.1 Peso Próprio e Empuxo

O peso próprio e o empuxo são considerados de forma conjunta, uma vez que deve ser fornecida como dado de entrada, para cada trecho de riser, a massa específica por unidade de comprimento do material que o compõe, bem como a densidade da água. Assim o cálculo

feito dentro do software fica extremamente simplificado e está explicitado na expressão (II.144). Calcula-se o empuxo como na equação e com a diferença obtém-se o peso efetivo do cabo submerso.

$$P = \rho \cdot L \cdot g \tag{II.144}$$

$$P = \rho_{agua} \cdot \pi \cdot D \cdot L \tag{II.145}$$

Onde: ρ é a massa específica por unidade de comprimento;

$\rho_{água}$ é a massa específica da água por unidade de comprimento;

L é o comprimento do trecho;

g é a aceleração da gravidade;

D é o diâmetro do cabo;

II.4.2 Corrente Marítima

A corrente marítima é fornecida através de perfis de corrente, tidos como entrada do programa. Usualmente se dispõe de uma tabela que relacione a cada profundidade com um valor de velocidade de corrente.

Para transformar esse perfil discreto em contínuo, é feita internamente ao programa uma interpolação linear a cada dois pontos fornecidos. Dessa forma, se obtém um perfil ilustrado na Figura 11.

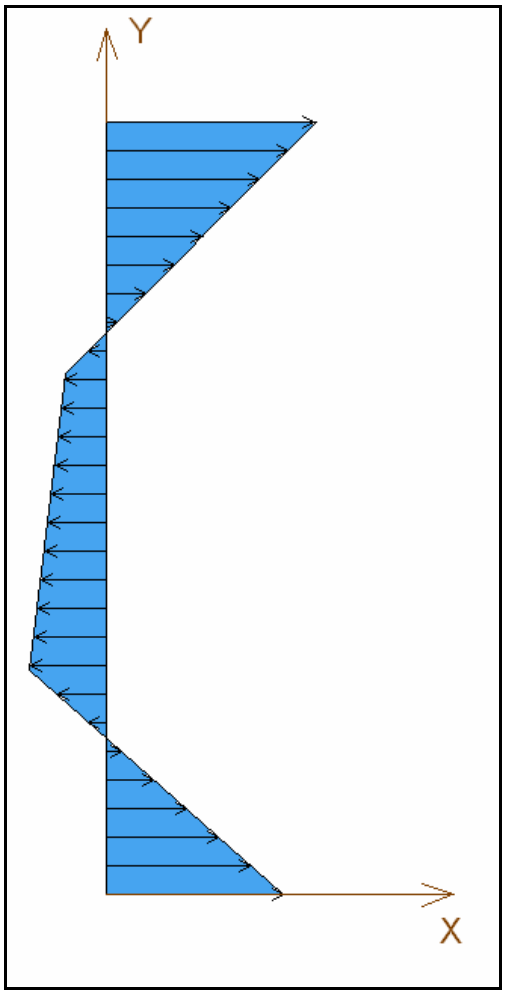


Figura 11 – Perfil de corrente obtido com interpolação linear entre as cotas

Fórmula de Morison

Para aplicar esse modelo, para cada elemento, é necessário realizar uma decomposição da velocidade, em duas direções: normal e tangencial à direção do elemento. Com posse dos coeficientes de arrasto nessas duas direções, tidos como dados de entrada, da massa específica da água do mar, e do diâmetro externo do riser, podem-se aplicar as equações (II.146) e (II.147).

$$\overrightarrow{F_n} = -\frac{1}{2}\rho_f D c_{d,n} \left|\vec{V}_n\right| \vec{V}_n \tag{II.146}$$

$$\overrightarrow{F_t} = -\frac{1}{2}\rho_f D c_{d,t} \left|\vec{V}_t\right| \vec{V}_t \tag{II.147}$$

Onde: $\overrightarrow{F_n}$ é a força de arrasto na direção normal;

$\overrightarrow{F_t}$ é a força de arrasto na direção tangencial;

ρ_f é a massa específica do fluido (água do mar);

D é o diâmetro externo;

$c_{d,n}$ é o coeficiente de arrasto na direção normal;

$c_{d,t}$ é o coeficiente de arrasto na direção tangencial;

$\vec{V}_n$ é o módulo da componente normal da velocidade;

$\vec{V}_t$ é o módulo da componente tangencial da velocidade.

II.5 Restrições ("Constraints") [9]

As restrições são formas de manter determinados graus de liberdade fixos. Como existem alguns tipos de restrições muito comum nas análises, foram criados tipos específicos de restrições. Engastes e articulações são exemplos. Porém, para haver maior flexibilidade, foi incorporada a possibilidade de o usuário definir exatamente a restrição que deseja. Para tal, é permitida a escolha de quais graus de liberdade terão seu movimento impedido.

II.5.1 Engaste

O engaste nada mais que o tipo de restrição que deixa todos os graus de liberdade nodais de rotação e translação fixos. Com essa restrição o nó é impedido de se movimentar tanto linear quanto angularmente e sofre a ação de algumas forças conhecidas como reações vinculares. Tais reações, para o engaste, compreendem forças e momentos.

II.5.2 Articulação

A articulação é o tipo de restrição que deixa todos os graus de liberdade nodais de translação fixos. Com essa restrição o nó é impedido de movimentar-se, porém é mantida a liberdade angular em todas as direções. As reações vinculares são somente forças, mas não momentos, como no caso do engaste.

II.5.3 Restrição definida pelo usuário

Nesse tipo de restrição, o usuário define quais graus de liberdade do nó estarão fixos e quais estarão livres. Como entrada para essa restrição, tem-se: o nó de atuação e os graus de liberdade fixos. Na entrada de dados, os graus de liberdade são representados por *X, Y, Z,* para translações nesses eixos e, *RX, RY,* e *RZ* para rotações em torno dos eixos *X, Y* e *Z* respectivamente.

Ainda existe a possibilidade de aplicar uma condição de contorno oblíqua em relação ao sistema de coordenadas global. Pode haver interesse por esse tipo de restrição por parte do usuário, em alguns problemas mais específicos.

II.5.4 Restrição do tipo Solo

Como exemplificado por Patel, 1995 [1], existem diversas formas de configurar um cabo submerso, cada qual apresentando suas peculiaridades. Para alguns tipos de configuração como, por exemplo, a “Simple-Catenary”, a “Lazy Wave” e a “Lazy-S”, existe um trecho do riser que fica acomodado sobre o fundo do mar e, em certo ponto o mesmo sai dessa posição e atinge cotas mais elevadas. Esse ponto é chamado de TDP. A Figura 12 mostra um exemplo de configuração de riser que contenha TDP.

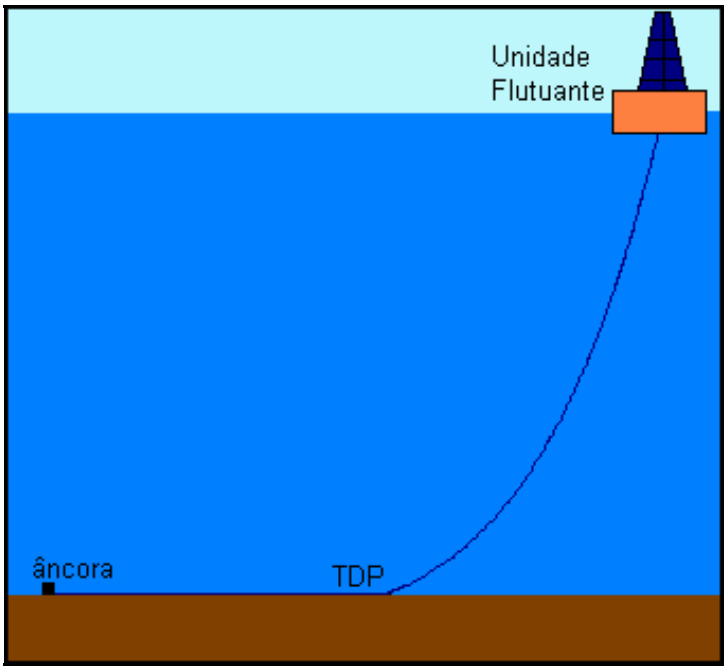


Figura 12 – Exemplo de Configuração do tipo "Simple Catenary" apresentando Touch Down Point (TDP)

A existência do contato entre o riser e o solo é um grande fator de geração de não linearidade no problema. A questão é que não se sabe, inicialmente, a posição do TDP e, portanto, não se sabe qual o comprimento de riser que estará apoiado no solo. Esse fato torna o problema muito mais complicado do que seria caso o ponto do TDP fosse conhecido. Necessariamente um processo iterativo é necessário para saber onde estará o TDP, dados os pontos extremos, a posição do solo e o comprimento do riser. Não se sabe, inicialmente, nem ao menos se irá ou não ocorrer contato com o solo, para cada condição particular.

Apesar da grande dificuldade esperada na resolução desse tipo de problema, é importante enfatizar sua importância, uma vez que a presença do solo pode mudar radicalmente a distribuição da tração no riser, em relação ao que seria se o mesmo fosse desconsiderado.

Para modelar a condição de contorno do solo é possível utilizar diversas abordagens. A primeira aproximação que pode, a princípio, ser pensada é a de considerar o solo como sendo infinitamente rígido e, portanto, nenhuma penetração do riser seria admissível.

Ainda é possível lidar com o problema pensando em outro tipo de abordagem: admitindo uma rigidez equivalente do solo, por unidade de área de contato, por exemplo. Dessa forma, estar-se-ia quantificando, a partir de um parâmetro físico, o grau de penetração admissível na cota onde se encontraria o solo antes do início do contato.

Também é possível tratar o problema admitindo penetrações no solo, segundo uma precisão estipulada e, aplicando penalidades às posições do riser que descessem abaixo da cota do solo. Definindo uma função objetivo visando minimizar essas penalidades seria possível transformar a questão proposta em um problema de otimização, que poderia ser resolvido por diversos métodos, tanto determinísticos quanto probabilísticos.

Nesse trabalho foi implementado o modelo do solo infinitamente rígido e a maneira de abordar a questão envolveu a imposição de deslocamentos em determinados graus de liberdade da malha, a fim de não violar a condição de contorno do solo. O algoritmo está explicado a seguir.

- ***Algoritmo***

Para atender à condição do contorno, são seguidos os seguintes passos na resolução do problema, a cada iteração:

É feita uma verificação em todos os nós presentes na malha, para verificar se os mesmos se encontram acima ou abaixo da cota do solo, que foi definida pelo usuário.

Para os nós que se encontram abaixo do solo, o grau de liberdade referente à translação na direção vertical é modificado para se tornar fixo, pois será imposto um deslocamento nessa direção em seguida.

Para cada nó que está abaixo do solo, é calculada a distância a ser imposta para cima, visando que o nó atinja exatamente a cota do solo. Essa distância é imposta no momento de resolver o sistema linear de cada iteração, através da técnica de condensação estática.

Após a imposição de deslocamentos, criou-se um desequilíbrio de esforços em todo o sistema. Dessa forma, para a próxima iteração os graus de liberdade que tinham sido marcados como fixos nessa iteração, por estarem abaixo do solo, serão marcados como livres, a fim permitir que o sistema "busque" sua posição de equilíbrio de forças nodais.

O processo é repetido a cada iteração, até que haja a convergência, tanto em equilíbrio de esforços como na não violação da condição de contorno do solo. Note que pode haver oscilações de posição de nós, entrando e saindo do solo, a cada iteração. Isso é esperado, visto que ocorrem, a cada iteração, mudanças nos tipos de graus de liberdade (fixos ou livres) dos nós que estão mais próximos do solo e que são afetados pelo algoritmo.

É importante enfatizar um detalhe quanto ao equilíbrio de esforços da estrutura: na implantação é admitido que o solo, quando em contato com a estrutura, responde com um esforço vertical na mesma, com a intensidade necessária para que haja equilíbrio nessa direção. Desconsidera-se, portanto, ações tangenciais ao solo da força de contato (atritos).

A Figura 13 ilustra a seqüência de funcionamento do algoritmo.

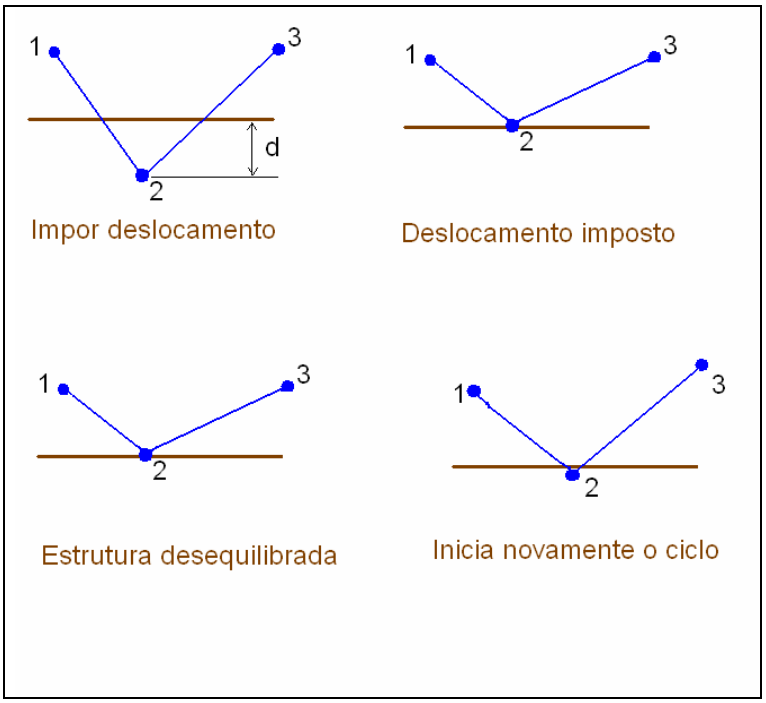


Figura 13 – Passos do algoritmo de imposição da condição de contorno do solo infinitamente rígido

Ainda que a idéia esteja correta e funcione para muitos problemas particulares, notou-se a existência de alguns problemas, nos quais ocorrem valores muito grandes de imposição de deslocamentos. Percebeu-se que ao tentar aplicar esse algoritmo nesses problemas, muitas vezes provocava-se um mal condicionamento matricial, de tal forma que ocorresse divergência numérica.

Para solucionar a questão, de maneira análoga ao método de aplicação de esforços de maneira incremental, criou-se a opção de impor cotas intermediárias do solo, a partir da posição do nó mais baixo, até a cota imposta pelo usuário para o solo.

Dessa forma, seria possível aplicar o algoritmo explicado nesse item diversas vezes, a cada nova posição intermediária imposta do solo. Note que a convergência tanto em termos de equilíbrio de esforços como em não violação da condição de contorno do solo deve ser requerida a cada incremento de cota do mesmo, a fim de evitar problemas de mal

condicionamento de matrizes causados por desequilíbrios nodais muito grandes ou, configurações que facilmente se tornem instáveis, o que muito provavelmente causaria a divergência do método.

- ***Condição de contorno oblíqua em relação ao sistema de coordenadas global***

A estratégia aplicada consiste em uma redefinição do sistema de coordenadas globais somente nos nós afetados pelas condições de contorno oblíquas, como pode ser visto em Cook [10].

Em termos práticos de aplicação do método, o que ocorre é que os graus de liberdade afetados por essas restrições devem ser rotacionados para novas direções, tais que haja alinhamento entre esse novo sistema de coordenadas global rotacionado e as direções de ação das reações dos vínculos oblíquos.

Os termos da matriz de rigidez global que estão associados aos graus de liberdade dessa restrição são recolhidos em uma matriz temporária. A seguir é aplicada uma rotação nessa matriz. Essa operação é definida pela inclinação da restrição em relação ao sistema global de coordenadas. A seguir, os termos rotacionados são re-alocados na matriz global.

É aplicada a mesma rotação nos esforços externos aplicados naquele nó, através de um vetor temporário, analogamente ao que se fez com a matriz de rigidez.

Agora é possível resolver o sistema global, e a restrição irá se comportar fisicamente na direção esperada.

A seguir se encontra uma forma para a matriz de rigidez global do sistema:

$$\mathbf{K_G} = \begin{bmatrix} k_{11} & k_{12} & \cdots & \cdots & \cdots & \cdots & \cdots & k_{1N} \\ k_{21} & k_{22} & \cdots & \cdots & \cdots & \cdots & \cdots & k_{2N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{(p-1)1} & k_{(p-1)2} & \cdots & k_{(p-1)(p-1)} & k_{(p-1)(p)} & k_{(p-1)(p+1)} & \cdots & k_{(p-1)N} \\ k_{(p)1} & k_{(p)2} & \cdots & k_{(p)(p-1)} & k_{(p)(p)} & k_{(p)(p+1)} & \cdots & k_{(p)N} \\ k_{(p+1)1} & k_{(p+1)2} & \cdots & k_{(p+1)(p-1)} & k_{(p+1)(p)} & k_{(p+1)(p+1)} & \cdots & k_{(p+1)N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{N1} & k_{N2} & \cdots & \cdots & \cdots & \cdots & \cdots & k_{NN} \end{bmatrix} \quad \text{(II.148)}$$

Supondo que as numerações *(p-1), (p) e (p+1)* representem os graus de liberdade associados a uma restrição inclinada, é definida uma matriz temporária (K_T) que armazene os coeficientes da matriz global dessas numerações.

$$\mathbf{K_T} = \begin{bmatrix} k_{(p-1)(p-1)} & k_{(p-1)(p)} & k_{(p-1)(p+1)} \\ k_{(p)(p-1)} & k_{(p)(p)} & k_{(p)(p+1)} \\ k_{(p+1)(p-1)} & k_{(p+1)(p)} & k_{(p+1)(p+1)} \end{bmatrix} \quad \text{(II.149)}$$

A matriz de rotação que será aplicada em K_T está ilustrada a seguir:

$$\mathbf{T} = \begin{bmatrix} \cos(x\bar{x}) & \cos(y\bar{x}) & \cos(z\bar{x}) \\ \cos(x\bar{y}) & \cos(y\bar{y}) & \cos(z\bar{y}) \\ \cos(x\bar{z}) & \cos(y\bar{z}) & \cos(z\bar{z}) \end{bmatrix} \quad \text{(II.150)}$$

Note que a matriz de rotação é composta pelos cossenos diretores entre os eixos coordenados final e inicial da rotação a ser feita.

Define-se uma matriz K_T' que representa a matriz K_T rotacionada – equação (II.151).

$$\mathbf{K_T}' = \mathbf{T}\mathbf{K_T}\mathbf{T}^{\mathbf{T}} \quad \text{(II.151)}$$

A seguir os termos de K_T' são re-alocados nas mesmas posições da matriz global K_G, de onde foram retirados os termos da matriz K_T.

Um procedimento análogo é feito com os vetores de esforços e após a resolução do sistema, com o vetor de deslocamentos livres da estrutura, obtidos na resolução do sistema linear.

II.6 Estrutura de Programação

O programa foi estruturado, a partir das necessidades dos dados anteriormente citados, de maneira a exigir a menor alteração de código possível para a inserção de um novo tipo de elemento ou restrição. Sendo assim, foram utilizados conceitos de abstração, herança e polimorfismo presentes no C++ [11] para a criação deste código portável e flexível. Para garantir fácil organização dos dados, foi utilizada uma padronização para o código, bem como foi escolhido um padrão de intercâmbio entre as demais partes do software (pré e pós-processador) que fosse de fácil organização e que pudesse ser visualizado igualmente em qualquer plataforma. Assim, escolheu-se o XML e foi feito o uso de uma biblioteca "freeware" que será discutida ainda nesse tópico com mais detalhes.

II.6.1 Linguagem e Padronização do Código

Como dito anteriormente, a linguagem utilizada foi o C++ por apresentar recursos como herança, polimorfismo, classes abstratas e puramente virtuais. Tais conceitos são os requisitos básicos para a estrutura delineada do programa.

Como a idéia central deste projeto é criar um ambiente flexível, com facilidade de expansão e com generalidade suficiente para que a inclusão de novas funcionalidades ficasse quase que restrita à geração de código referente àquela classe, com poucas ou se possível nenhuma alteração no restante do código, o conceito de classes virtuais e herança é muito utilizado. Por exemplo, cria-se então uma classe virtual de elemento e derivam-se classes dessa, específicas

para cada tipo de elemento (Ex: Treliça, Cabo). Utilizando-se de uma estrutura de lista duplamente ligada, cria-se uma "lista de elementos" independentemente de quais estes sejam. Assim a inclusão de um novo elemento na leitura e a inclusão da classe do mesmo permitirá que este seja facilmente incorporado ao projeto.

A estrutura de lista ligada descrita anteriormente é basicamente composta de um elemento denominado *m_first* do tipo da lista em questão, de um *m_last* que determina o elemento final da lista e o número de nós *m_nnos*. Cada elemento da lista possui uma variável *m_next* que aponta para o elemento seguinte da lista e uma variável *m_previous* que aponta para o elemento anterior, o que dá origem ao nome, lista duplamente ligada, uma vez que pode ser percorrida por ambas as extremidades. Assim inserem-se elementos na lista apenas efetuando as ligações entre os elementos que a compõem. Trabalhou-se também na formulação de uma padronização do código, visando tanto a melhor compreensão do mesmo quanto uma futura expansão. A padronização segue a seguinte estrutura:

- Nomes de Arquivos: todas as letras em minúsculas.

  Exemplos: *element.h*, *element.cpp*

- Nomes de Classes: começam com uma letra maiúscula C e o nome da Classe (Letras maiúsculas para início dos nomes)

  Exemplo: *CElement*

- Nomes de Variáveis membros de classe: começam com uma letra minúscula *m* e um caractere "underline" (todas as letras em minúsculas).

  Exemplo: *m_next*

- Para variáveis globais, convencionou-se trocar o *m* por um *g*, facilitando assim a identificação.

Exemplo: *g_free_DOF*

- Para variáveis de leitura do XML, já que derivam de uma biblioteca "freeware" adotou-se iniciar com um *x*.

Exemplo: *x_element*

- Diretórios: Arquivos de diferentes tipos vão para diretórios diversos.

Exemplo: Todos os elementos ocupam o diretório *elements* (todas as letras em minúsculas), assim, devemos incluir *".\elements\element.h."*

II.6.2 Descrição das Classes

Por se trabalhar com os conceitos anteriormente citados, criou-se uma estrutura de classes tal que permita toda a generalidade e flexibilidade buscada. Na Tabela 5 descrever-se-á as classes, classificadas alfabeticamente.

Tabela 5 – Classes que compõe o programa

Classe	**Descrição**
CBandMatrix	Responsável pelo armazenamento das matrizes na estrutura de banda. Possui funções para armazenar os valores na matriz, lê-los, determinar as coordenadas na estrutura de banda e na estrutura quadrada (conversão entre essas) e funções de acesso às dimensões da matriz.
CCable	Implementa o elemento finito do tipo cabo. Derivada da *CElement*, implementa as funções da mesma para o tipo de específico de elemento, ou seja, determina as matrizes de rigidez e rotação específicas, dadas de acordo com as propriedades do mesmo e dos nós a que o mesmo está associado (no caso, dois).
CCase	Classe puramente virtual que possui todas as rotinas para rodar-se um caso no programa. A sua estrutura virtual é explicada para abrigar novos casos sem grandes alterações estruturais no código. Só foi desenvolvido o caso estático, descrito adiante, por ser o escopo do projeto, mas esse foi projetado de tal forma a permitir inclusões futuras, como, por exemplo, o caso dinâmico.
CConstraints	Classe virtual que dá as diretivas de como serão as restrições. Possui a descrição das funções de montagem de conexão, leitura e gravação em XML e parâmetros relativos a estas restrições, como por exemplo, o número de nós e quais os nós associados ao vínculo em questão.

continua

Classe	Descrição
CConstraintList	Classe que implementa uma lista ligada de vínculos. A lista ligada possui sempre a mesma estrutura em todo o programa, como já foi descrito.
CCurrent	Classe que define um perfil de corrente. Possui a profundidade na qual esse perfil está atuando e a velocidade da corrente nesse perfil. A direção de atuação da corrente (ângulo azimutal) é dada aqui e é igual para uma mesma corrente.
CCurrentList	Classe que implementa a corrente. Nessa classe ainda é possível obter-se o valor da corrente em pontos intermediários graças a interpolação linear por ela feita. Essa classe não possui variáveis.
CDOF	Classe que armazena os tipos e numeração dos graus de liberdade. Uma instância dessa classe está sempre associada a um nó, sendo assim utilizada para determinar em quais posições as contribuições de cada grau de liberdade na matriz global de rigidez serão somadas.
CEnvironment	Classe que armazena os dados do ambiente no qual o objeto de estudo está inserido. As propriedades são, aceleração da gravidade, densidade da água e comprimento da lâmina d'água.
CElementsList	Classe que implementa uma lista ligada de elementos.

continuação

Classe	Descrição
CElement	Classe virtual de elementos. Nessa classe estão definidas as propriedades características de cada elemento, como por exemplo, a matriz de rigidez local, número de nós associados ao elemento, tipo do elemento, as bandas das matrizes K_{AA} e K_{BB} (ver condensação estática) de cada elemento, a matriz de rotação, bem como funções para determinar as bandas da mesma, para montar as matrizes local, global, de conexão e de rotação, entre outras.
CFixed	Classe que implementa um tipo de restrição – o engaste. Derivada de *CConstraint*, implementa as funções da mesma para o tipo de específico de restrição, ou seja, restringindo todos os graus de liberdades livre no nó ao qual esta restrição está associada.
CGlobalLists	Classe na qual estão as listas globais do programa, com exceção da lista de nós que, por ser acessada por todas as partes do programa, está como global. Possui como variáveis principais a lista de esforços, a de elementos e a de restrições.
CJoint	Classe que implementa um tipo de restrição – a articulação. Derivada de *CConstraint*, implementa as funções da mesma para o tipo de específico de restrição, ou seja, restringindo todos os graus de liberdades de translação livre no nó ao qual esta restrição está associada.

continuação

Classe	Descrição
CGlobalVariables	Classe que armazena todas as variáveis globais do programa, com exceção das listas, que estão na classe global de listas. Tal organização torna o programa mais seguro, dado que o acesso, mesmo sendo global, é mais restrito por ser utilizado um conjunto de *gets* e *sets*. Possui como variáveis principais as matrizes globais e os valores de bandas globais, além do ambiente em questão.
CProperty	Classe que armazena todas as informações relativas a uma propriedade específica, como sua rigidez axial (*EA*) e rigidez torcional (*GJ*), além de dados como os coeficientes de arrasto (tanto normal quanto tangencial) e diâmetro hidráulico.
CPropertiesList	Classe que implementa uma lista ligada de propriedades.
CMatrix	Classe que armazena a estrutura de uma matriz de dimensões *m* e *n* dados pelo usuário, de números reais, com os respectivos operadores. Pertence a um pacote de funções criado para auxiliar o desenvolvimento.
CNodalLoad	Classe que implementa os carregamentos nodais, composta basicamente de um vetor e de funções de leitura e gravação tanto para uso do programa quanto no XML, como as demais classes que armazenam dados do programa.

continuação

Classe	Descrição
CNodalLoadList	Classe que implementa uma lista ligada de carregamentos nodais.
CNode	Classe que armazena as coordenadas dos nós, bem como as funções de leitura, gravação em XML e também no programa.
CNodeList	Classe que implementa uma lista ligada de nós.
CParameters	Classe que armazena os parâmetros a serem utilizados na simulação, como número de iterações, precisão e método a ser utilizado.
CReadWrite	Classe responsável por toda operação de *IO* com os arquivos XML. Essa classe chama todas as leituras e gravações das outras classes, permitindo assim uma maior garantia quanto aos dados de entrada e saída, uma vez que o controle dessas operações fica facilitado.
CRestaurationIncremental	Classe que implementa o método dos esforços restauradores – Método de Newton-Raphson – de maneira incremental, ou seja, ele atinge o equilíbrio em várias posições intermediárias, com carregamentos menores, até chegar na configuração final.

continuação

Classe	Descrição
CRestaurationMethod	Classe que implementa o método dos esforços restauradores – Método de Newton-Raphson – de maneira direta, utilizando-se do conceito de esforço desbalanceado, que é explicado nesse documento.
CRun	Classe puramente virtual do método que será utilizado na análise. Cada método deve ser escolhido de acordo com o problema a ser resolvido. Alguns exemplos de classes derivadas de *CRun* são a classe *CRestaurationMethod* e *CTangentStiffness*.
CSolver	Classe composta por diversas funções destinadas a resolução matemática, como por exemplo, funções para determinar soluções de sistemas lineares (Cholesky, LDLT, Gauss, entre outras formas), funções para cálculo de multiplicações entre vetores e matrizes específicas. Algo a se considerar é que a maioria das funções está implementada considerando a estrutura de banda utilizada, otimizando assim a performance dos métodos.
CStaticCase	Classe derivada de *CCase* que é responsável pela instanciação do caso estático, por chamar todas as funções necessárias para criar o caso e rodá-lo e ainda por resolver o sistema final, retornando assim a configuração deformada desejada.

continuação

Classe	Descrição
CTangentStiffness	Classe que implementa o método de resolução do problema não-linear através de aproximações sucessivas da rigidez nos pontos em questão.
CTruss	Classe que implementa um tipo de elemento – a treliça. Derivada da *CElement*, implementa as funções da mesma para o tipo de específico de elemento, ou seja, determina as matrizes de rigidez e rotação específicas, dada de acordo com as propriedades do mesmo e dos nós que o mesmo está associado (no caso, dois).
CUserDefined	Classe que implementa um tipo de restrição – a definida pelo usuário. Derivada de *CConstraint*, implementa as funções da mesma para o tipo específico de restrição, ou seja, restringindo todos os graus de liberdades livres determinados pelo usuário na hora da entrada de dados (feita pelo XML), no nó ao qual esta restrição está associada.
CVector	Classe que armazena a estrutura de um vetor de dimensão *n* dado pelo usuário, de números reais, com os respectivos operadores. Pertence a um pacote de funções criado para auxiliar o desenvolvimento

conclusão

II.6.3 Leitura e Escrita em XML

Tanto a leitura e gravação são feitas através de arquivos no formato XML por apresentar vantagens em relação a um arquivo de texto comum. Percebe-se imediatamente no XML um tipo de estruturação nos dados dada pelas *tags*, uma vez que este se assemelha muito a um código em HTML, com a ressalva de que as *tags* também são definidas pelo programador, muito semelhante à estrutura de listas ligadas no programa. A Figura 14 mostra um trecho extraído de um XML de entrada do programa para melhor exibir esta concepção.

```
<Nodes>
    <Node x=" 0" y = "0" number="1">Origin</Node>
    <Node x="10" y = "0" number="2"></Node>
    <Node x="20" y = "0" number="3"></Node>
</Nodes>
```

Figura 14 – Exemplo de entrada do XML (parcial)

Assim, criando uma estruturação também na leitura, poder-se-á ler os nós do XML em qualquer ordem. Um outro aspecto importante é que, como pode ser visto no exemplo anterior, não há coordenada z, mas, devido a uma inicialização adequada das variáveis, essa não faz falta, caracterizando um problema plano. Assim percebe-se que, mesmo que o XML não possua todos os parâmetros esse não gerará erros de leitura. O único fato que deve-se atentar é que se alguns dados imprescindíveis à simulação estiverem faltando, apesar de não gerar erros na leitura, não será possível efetuar-se a simulação. Os nós do arquivo XML necessários e suportados pelo programa são descritos na Tabela 6**.**

Tabela 6 - Descrição dos nós do XML

Nó	Descrição
Ultra	Nó principal – raiz todos os outros devem estar contidos internamente a este
Properties	Nó no qual devem estar contidos todos as propriedades
Propertie	Nó que contém um material. Contém os atributos: *EA*, *EI*, *GJ* , HidraulicD, TangentialDrag NormalDrag e *Number*.
Environment	Nó que contém o ambiente. Contém os atributos: *Gravity*, *Water_ro*, *Water_depth e Seabed.t*
Parameters	Nó que contém os parâmetros de simulação. Contém os atributos: *Precision*, *MaxIterations*, *Method*, *Analyses*, *Steps*, *DWeight, DSoil, DCurrent* e *System*.
Nodes	Nó no qual devem estar contidos todos os nós
Node	Nó que contém um nó da malha. Contém os atributos: *X, Y, Z, RX, RY, RZ* e *Number*
Element	Nó no qual devem estar contidos todos os elementos
Truss, Cable, Beam	Tipo de elemento. Pode ou não estar presente. No caso de um elemento *truss*, *cable ou beam*, deve estar contido no nó *Element*. Contém os seguintes atributos: *Number, Node1, Node2, Property, PreTension* e *Tension*

continua

Nó	Descrição
Current	Nó no qual é definida a cota e a velocidade da corrente nesta. Possui os seguintes atributos: *Number*, *Depth, Direction* e *Velocity*.
CurrentList	Nó no qual devem estar contidos todos os perfis de corrente. Possui os seguintes atributos que são característicos do perfil:
Constraints	Nó no qual devem estar contidos todas as restrições
Fixed	Tipo de restrição. Pode ou não estar presente. No caso de uma restrição *fixed*, deve estar contida no nó *constraint*. Contém os seguintes atributos: *Number* e *Node*
Joint	Tipo de restrição. Pode ou não estar presente. No caso de uma restrição *joint*, deve estar contida no nó *constraint*. Contém os seguintes atributos: *Number* e *Node*
UserDefined	Tipo de restrição. Pode ou não estar presente. No caso de uma restrição *user defined*, deve estar contida no nó *constraint*. Contém os seguintes atributos: *Number , Node* e *DOF,* no qual são descritos os graus de liberdade a serem restritos. Ex: "*X Y RZ"* que restringe *X, Y* e a rotação em *Z*.

continuação

Nó	Descrição
NodalLoads	Nó no qual devem estar contidos todos os carregamentos nodais
NodalLoad	Nó que contém um carregamento nodal. Contém os atributos: *Fx, Fy, Fz, Mx, My, Mz* e *Node*

conclusão

Além disso, é necessário um cabeçalho indicando que se trate de um arquivo XML, como por exemplo, o exibido na Figura 15.

```
<?xml version="1.0" ?>
<!-- Arquivo de Entrada  -->
```

Figura 15 – Cabeçalho do arquivo XML de entrada

Quando se trata de um arquivo de saída, este cabeçalho é gerado automaticamente pelo programa.

II.6.4 Bibliotecas externas

A única biblioteca externa utilizada nesta etapa de projeto foi a biblioteca "*opensoure*" denominada TinyXML, e está disponível para "*download*" em *SourceForge* (http://sourceforge.net), a qual permite a leitura do XML através da criação de elementos do

tipo XML como nós, muito similar a uma estrutura de dados em árvore. A gravação do XML ocorre da mesma maneira, porém de maneira inversa na árvore, uma vez que a leitura é feita primeiro nos nós mais extremos enquanto a gravação é feita a partir do nó raiz e posterior inserção deste no arquivo de saída.

II.7 Simulações Realizadas

Com o objetivo de validar o método não linear desenvolvido, bem como realizar análises lineares de MEF utilizando a estruturação empregada no software, foram feitas algumas simulações a respeito de certas geometrias de riser, das quais eram previamente conhecidas as soluções analíticas.

O programa é capaz de solucionar casos em que não existe solução analítica, como configurações que sofrem efeito de corrente marítima, porém na fase inicial de testes, foram feitas comparações com casos em que é possível se determinar analiticamente a solução esperada, para em seguida comparar-se casos reais com programas comerciais.

II.7.1 Cabo bi-apoiado

Uma possível configuração de riser para aplicações na indústria offshore, utilizada quando se deseja interligar duas unidades flutuantes, é a forma de cabo bi-apoiado, submetido apenas ao carregamento do peso próprio. Fixando-se, como condições de contorno, os dois extremos do cabo, e entrando com seu comprimento, pode-se calcular analiticamente a tração em toda sua extensão. Tal cálculo foi feito desconsiderando-se os efeitos de corrente marítima.

Alternativamente, pode-se realizar uma discretização no sistema, dividindo-o em elementos de treliça ou cabo, e utilizar o software em desenvolvimento para determinar a geometria final, bem como a tração em todos os elementos.

Foi feito um primeiro teste para um caso particular cujo comprimento do cabo é de $L = 970{,}2\,\mathrm{m}$, a massa específica do material é de $7{,}5\mathrm{Kg/m}$ e os pontos de fixação extremos do cabo são:

A = (0 , 241 , 0)

B = (246 , 734 , 0)

Adotou-se como malha inicial uma discretização da própria geometria da solução analítica, e, o papel do software foi apenas de determinar corretamente a tração distribuída no cabo. Realizando uma discretização com 491 elementos, foram realizadas simulações utilizando elementos de treliça e de cabo. Os resultados estão mostrados a seguir, da Figura 16 à Figura 19.

Nota-se nos gráficos apresentados que a solução analítica e a obtida por MEF não linear (com elementos de Treliça e Cabo) praticamente se sobrepuseram, tanto na geometria da linha elástica como na tração ao longo do cabo. Como na solução analítica não existem elementos, o cálculo da tração para comparação com o resultado numérico foi feito nos pontos médios entre os nós. Esse procedimento também foi realizado na simulação posterior.

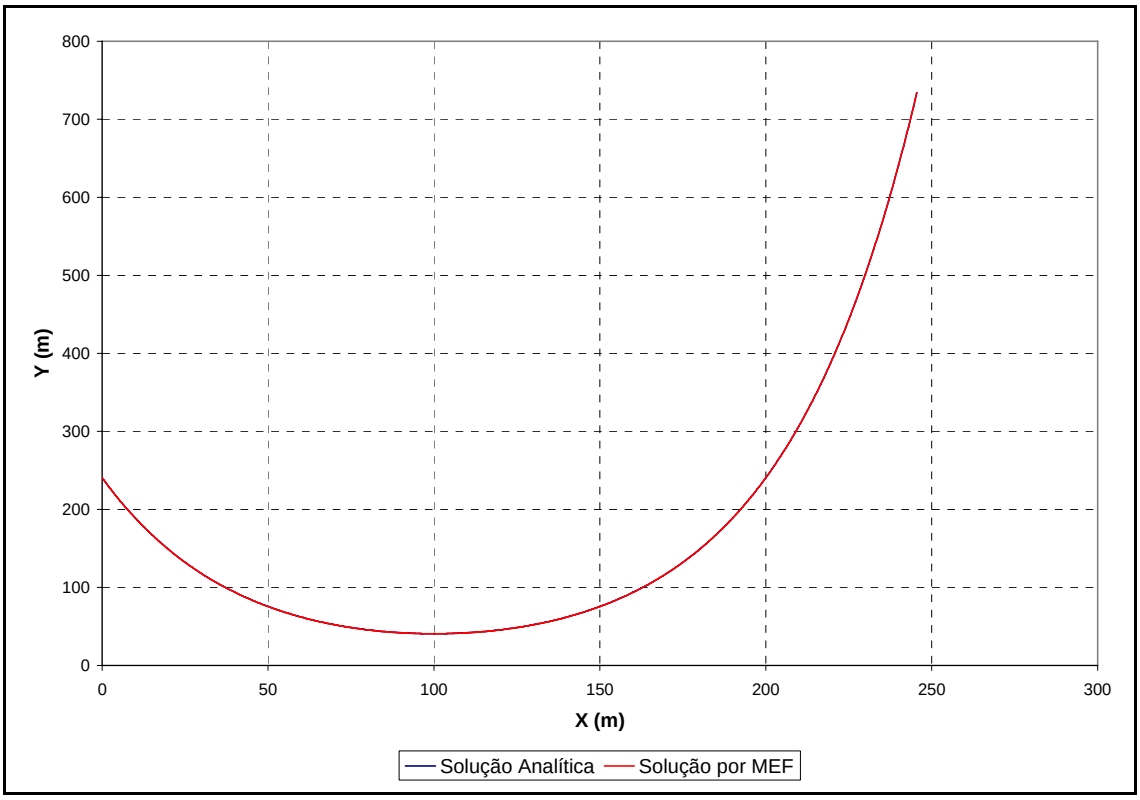


Figura 16 – Linha Elástica do caso Cabo bi-apoiado: Simulação realizada utilizando elementos de Treliça

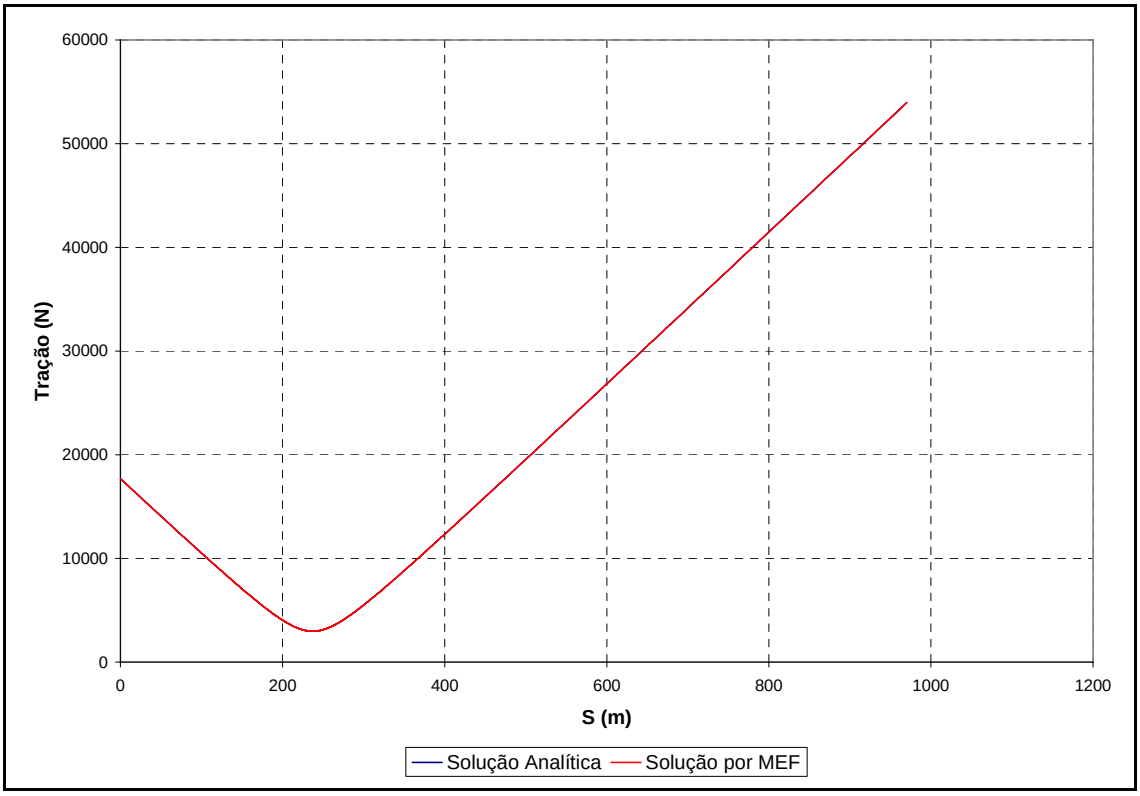


Figura 17 – Tração vs. Abscissa Curvilínea S do caso Cabo bi apoiado – Simulação realizada utilizando elementos de Treliça

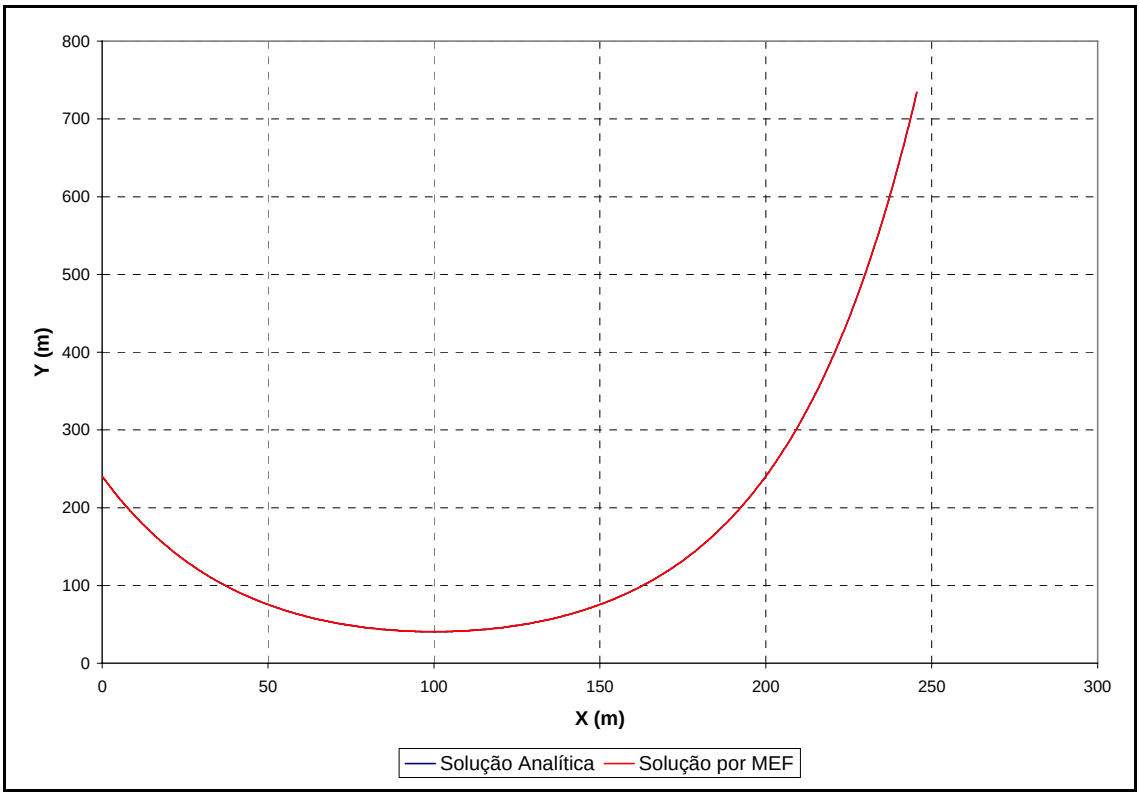


Figura 18 – Linha Elástica do caso Cabo bi-apoiado – Simulação realizada utilizando elementos de Cabo

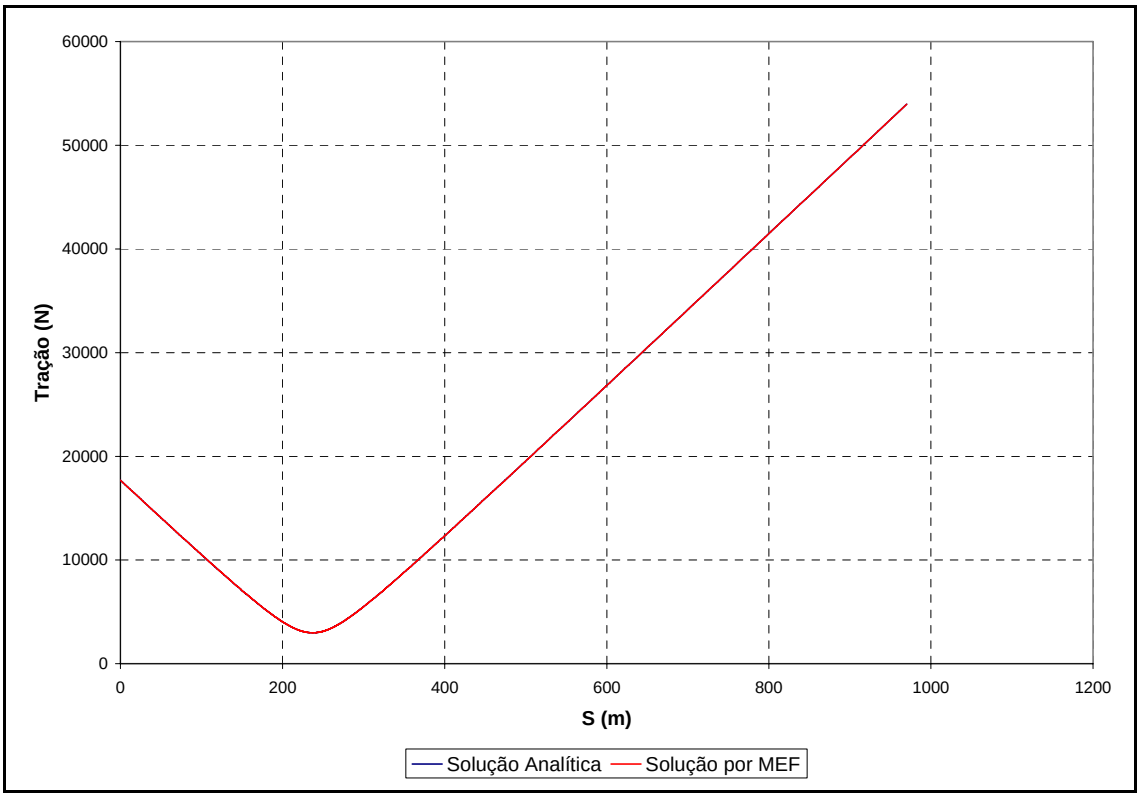


Figura 19 - Tração vs. Abscissa Curvilínea S do caso Cabo bi apoiado – Simulação realizada utilizando elementos de Cabo

II.7.2 Lazy-Wave sem "touchdown point"

Em muitas aplicações, risers são utilizados para interligar a unidade flutuante à âncora no fundo do mar. Quando a tração no topo do riser se torna muito alta, ao utilizar uma configuração do tipo catenária, por exemplo, pode se tornar viável a utilização de flutuadores fixados em certo comprimento do riser, proporcionando a redução na tração do topo. Com essa inserção, surge a configuração Lazy-Wave. Também determinável analiticamente quando submetido apenas ao peso próprio, sem correntes marítimas, um caso de Lazy-Wave foi simulado com elementos de treliça, com as seguintes características:

Trecho 1: $L_1 = 350{,}0\,\text{m}$ $P_1 = 47.775N$

Trecho 2: $L_2 = 378{,}3\,\text{m}$ $P_1 = -27.805N$

Trecho 3: $L_3 = 472{,}1\,\text{m}$ $\rho_3 = 34.699N$

Coordenada da âncora: *A = (0 , 0 , 0)*

Coordenada do topo: *B = (558 , 784 , 0)*

Discretização para solução por MEF*: 658* elementos

Exibe-se além dos valores dos comprimentos o peso efetivo do cabo, calculado pela diferença entre o peso do cabo e o empuxo exercido pela água. Os resultados obtidos com esse conjunto de dados foram os seguintes:

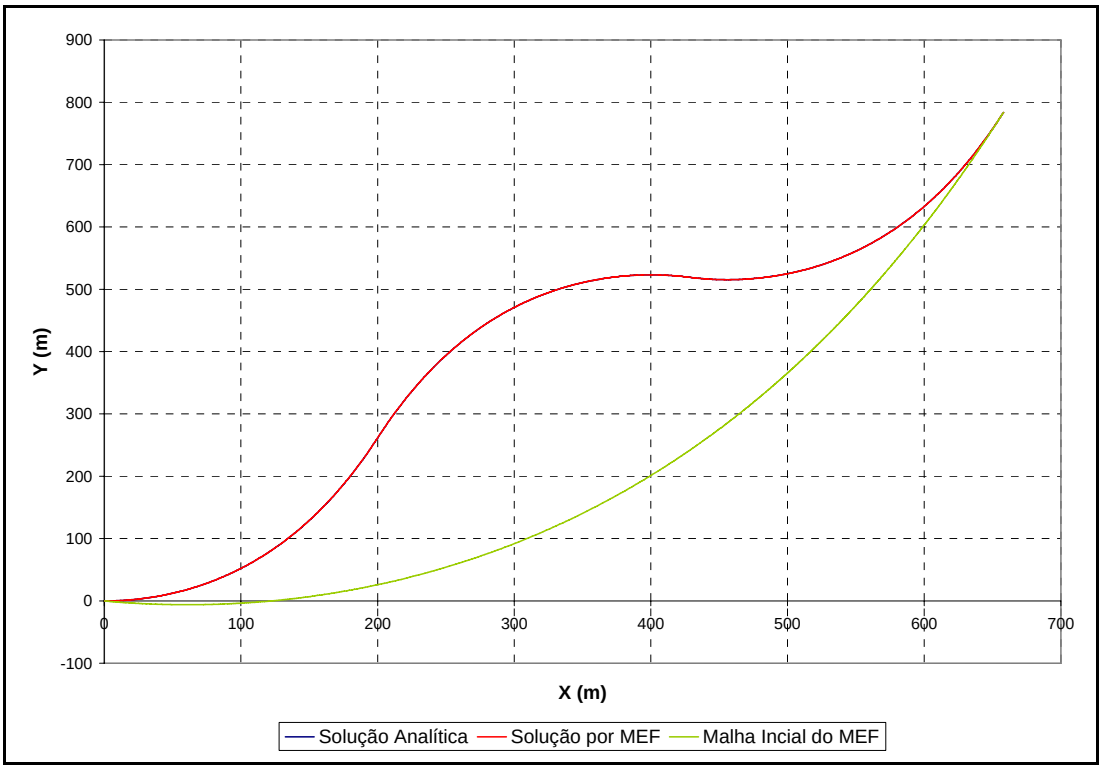


Figura 20 – Linha Elástica do caso Lazy-Wave – Simulação realizada utilizando elementos de Treliça

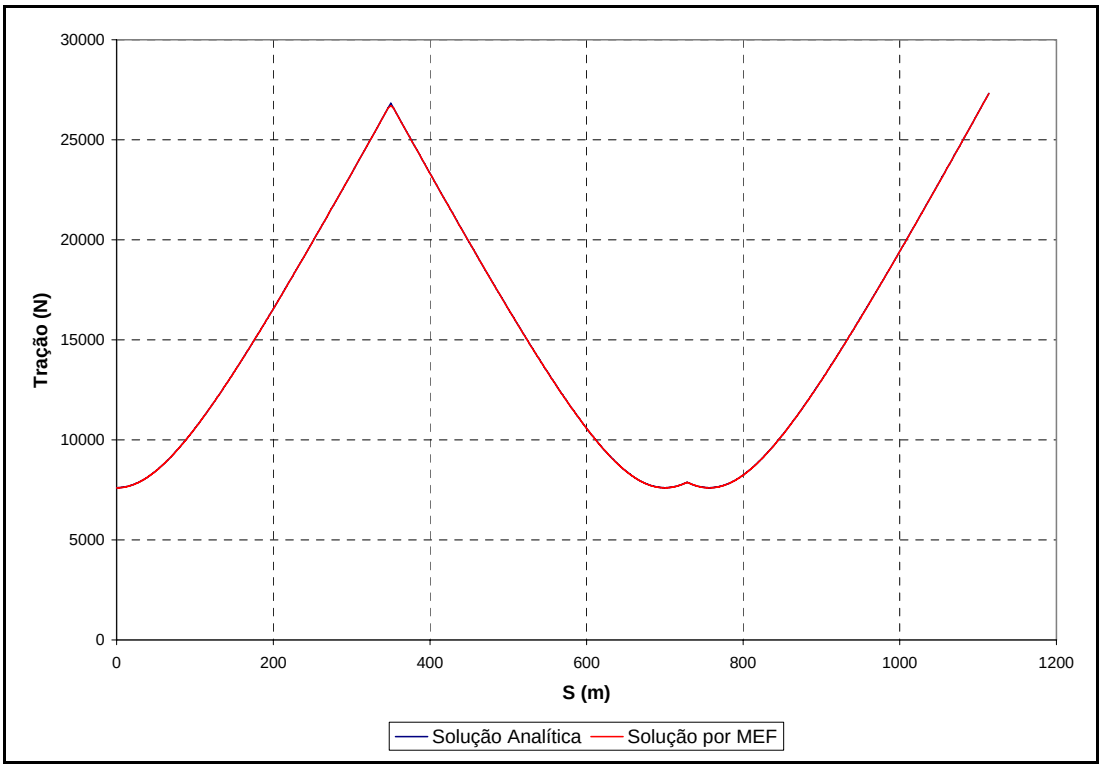


Figura 21 – Tração vs. Abscissa Curvilínea S do caso Lazy-Wave – Simulação realizada utilizando elementos de Treliça

Note na Figura 20 que desta vez a malha inicial utilizada para simular o caso com o MEF não possui a mesma geometria da configuração analítica. Portanto, exigiu-se do programa não só o ajuste da tração, mas também grandes deslocamentos de nós, de maneira a atingir a geometria esperada como solução. Tanto a geometria determinada, como a tração distribuída estão muito próximas da solução analítica.

II.7.3 Riser híbrido auto sustentável (RHAS)

O riser híbrido auto sustentável (RHAS) é uma configuração alternativa de riser, que consiste em um riser rígido vertical, com um "jumper" flexível na sua extremidade superior, e fixado no outro extremo em uma unidade flutuante (Figura 22).

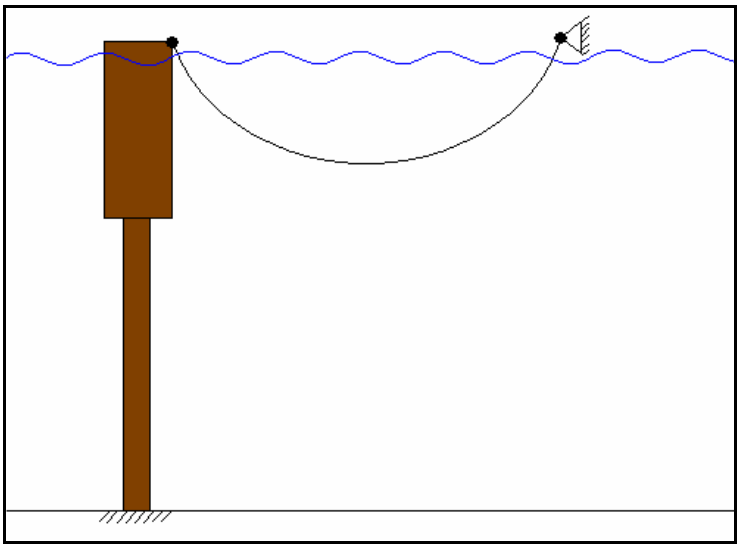


Figura 22 – Riser Híbrido Auto Sustentável

O modelo utilizado para simular o jumper do RHAS consiste em admitir uma rigidez equivalente de flexão do topo da estrutura do riser vertical. A Figura 23 ilustra esse modelo.

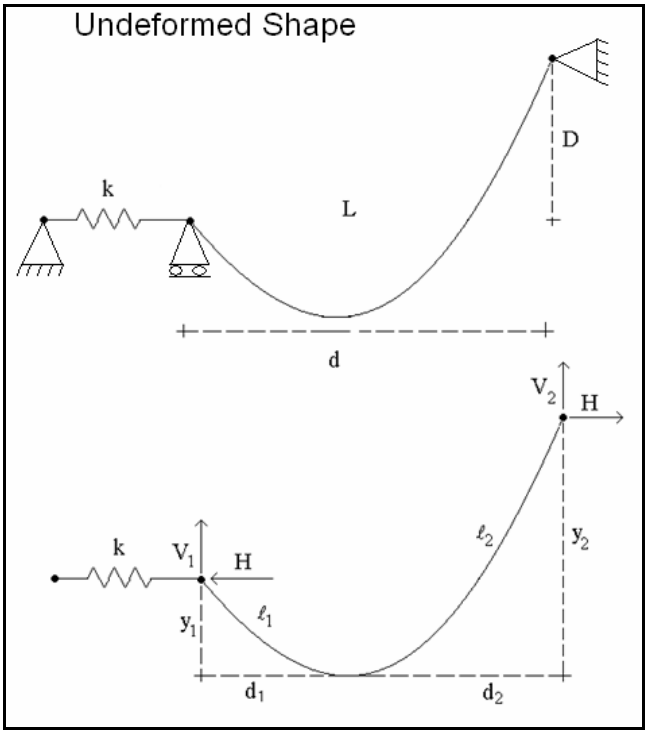


Figura 23 – Modelo adotado na simulação do RHAS

O modelo proposto possui solução numérica obtida com auxílio de um software comercial, com a qual foi comparado o resultado obtido pela simulação realizada utilizando MEF. Os dados de entrada estão descritos a seguir:

D = 100 m

d = 100 m

L = 386m

K = 1.9 N/m

$\rho = 7{,}5 Kg/m$

A Figura 24 ilustra a linha elástica determinada por simulação utilizando MEF. Note que o caso testado possui um valor de K (constante da mola de rigidez equivalente de flexão) muito pequeno. Por isso, houve um deslocamento tão grande (mais de 150 metros) do extremo do cabo situado na articulação com rodas.

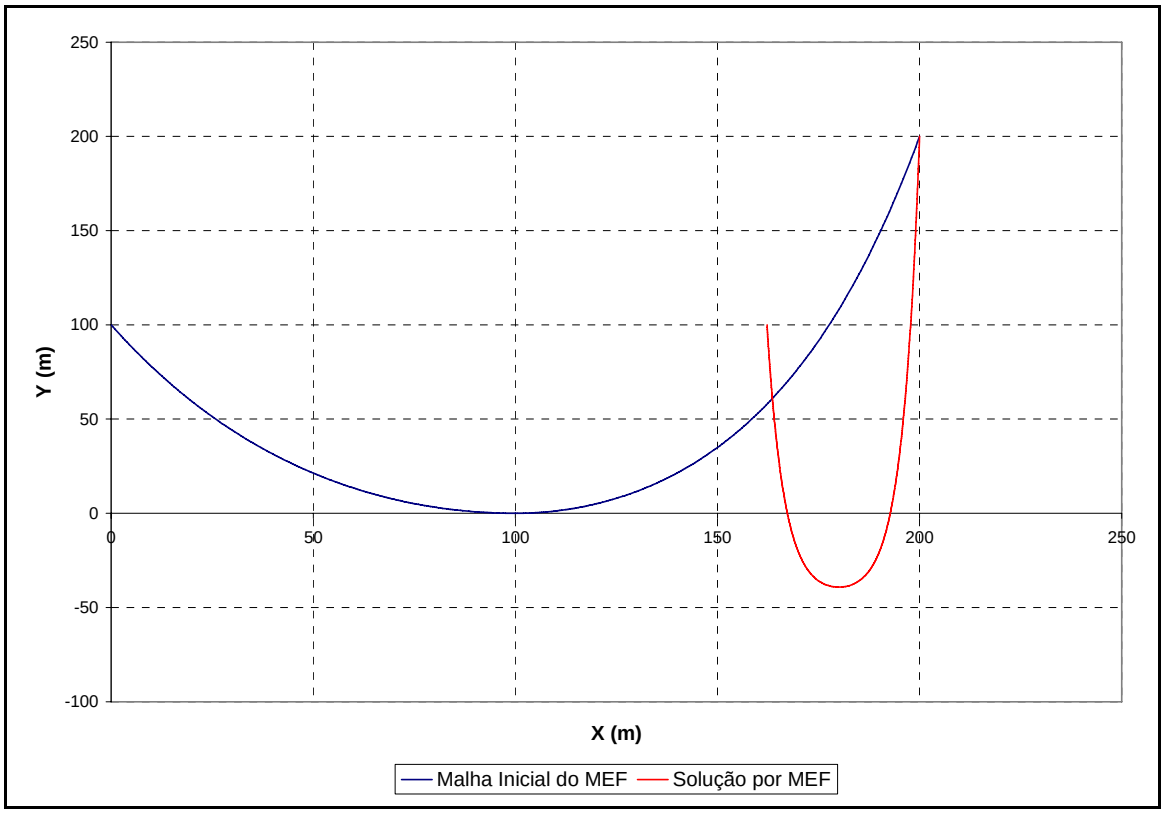


Figura 24 – Linha Elástica do RHAS simulado com elementos de treliça

Foi feita uma comparação de resultados obtidos na simulação utilizando MEF e na solução analítica.A Tabela 7 mostra alguns valores comparativos.

Tabela 7 – Resultados comparativos do caso RHAS

Pontos Mínimos da Linha Elástica			
	X (m)	Y (m)	Z (m)
MEF	180.247	-39.1854	0
Sol. Analítica	179.902	-39.1814	0
Força na Mola			
MEF	309.3348 N		
Sol. Analítica	309.3356 N		

Nota-se que os resultados comparativos entre MEF e solução analítica estão muito próximos.

II.7.4 Configuração Lazy Wave com touch-down point

Uma vez realizadas algumas simulações para comparação com soluções analíticas, também foram montados casos de mais difícil convergência numérica, como é o caso da imposição do contato unilateral com o solo. Assim, foram realizadas duas simulações de um caso Lazy-Wave com contato unilateral com o solo. Os dados do problema estão descritos a seguir:

Pontos extremos: A(0,0,0)

B(2340,0,1255)

Cota do solo: z = 0 m

Cota da superfície da água: z = 1255 m

Densidade da água: 1025 Kg/m3

Aceleração da gravidade: g = 9,807 m/s2

Número de segmentos: 3 segmentos

Propriedades dos segmentos (de A até B) – Exibido na Tabela 8.

Tabela 8 – Propriedades do riser simulado de uma configuração Lazy-wave

Segmento	**1**	**2**	**3**
Diâmetro (m)	0,4572	1,137	0,4572
Peso no ar (KN/m)	3,4	3,4	3,4
EI (KNm2)	167450	167450	167450
GJ (KNm2)	334900	334900	334900
EA (KN)	7159999	7159999	7159999
Comprimento (m)	1075	377	1591

Foram feitas duas simulações para comparar com outros programas comerciais:

a) Não considerando a rigidez flexional (elementos de treliça)

b) Considerando a rigidez flexional (elementos de pórtico)

Os resultados das linhas elásticas estão mostrados na Figura 25 e na Figura 26.

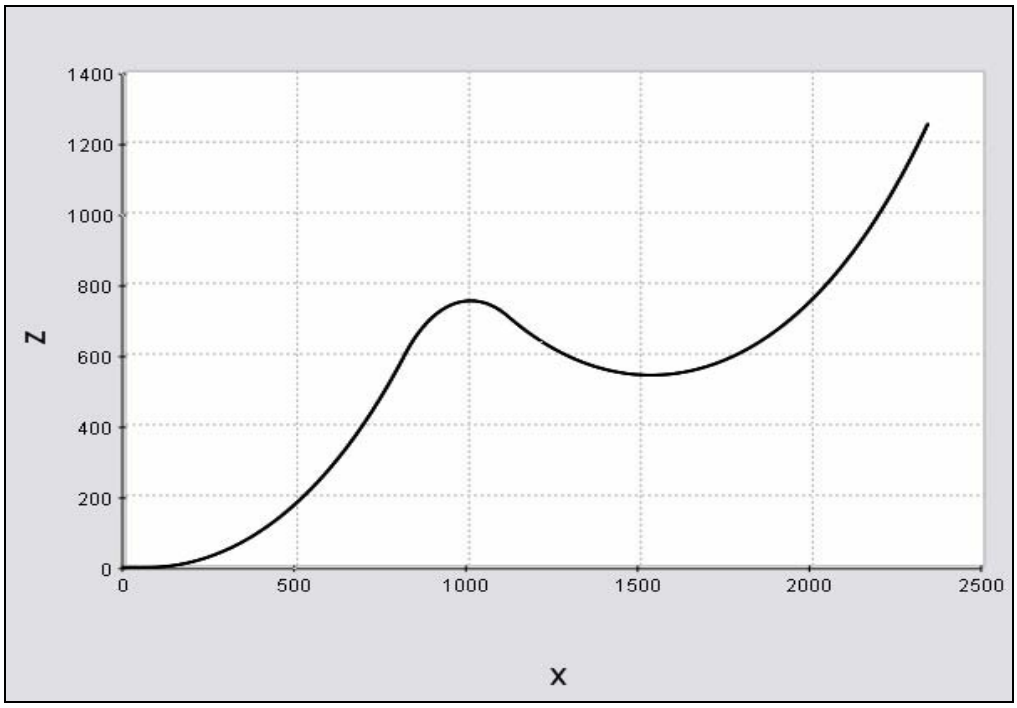


Figura 25 – Linha elástica de configuração Lazy-Wave simulada com elementos de treliça com touch-down point (eixos X e Z em metros)

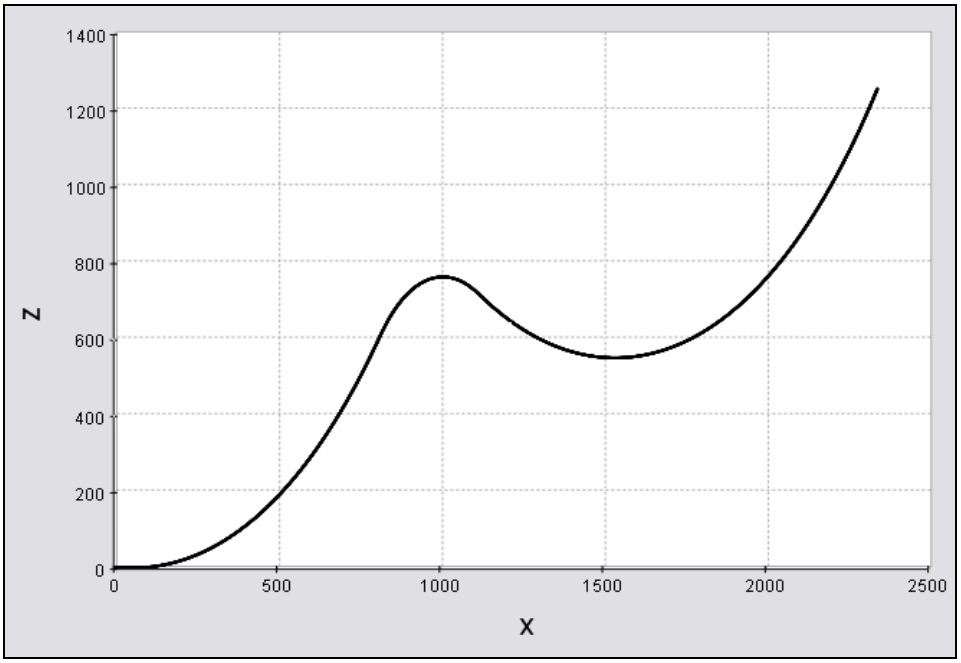


Figura 26 - Linha elástica de configuração Lazy-Wave simulada com elementos de pórtico com touch-down point (eixos X e Z em metros)

Alguns valores numéricos comparativos obtidos estão mostrados na Tabela 9 e na Tabela 10.

Tabela 9 - Alguns resultados comparativos para a configuração Lazy-Wave com touch-down point simulada com elementos de treliça

	OrcaFlex®	PoliFlex	**UltraFlex**
Tração no topo (KN)	2199,3104	2199,0939	**2195,947**
Posição do TDP (m)	67	66,36	**68,38**
Tração no TDP (KN)	956,67	955,57	**955,38**

Tabela 10 - Alguns resultados comparativos para a configuração Lazy-Wave com touch-down point simulada com elementos de pórtico

	OrcaFlex®	PoliFlex	**UltraFlex**
Tração no topo (KN)	2197,255	2199,039	**2193,805**
Posição do TDP (m)	56	53,13	**56,086**
Tração no TDP (KN)	952,9714	956,9669	**959,265**

Os resultados se mostraram compatíveis com os outros dois programas de computador, que possuem métodos diferentes de análise, servem de comparação para validação dos modelos implantadas no software desenvolvido.

II.8 Conclusão

Em todas as simulações realizadas e que foram comparadas com soluções analíticas houve ótimos resultados. Isso leva à conclusão da eficiência do método não-linear empregado, mesmo quando foi mais exigido, como é o caso das simulações Lazy-Wave, na qual a geometria da malha inicial é bastante distante da solução final esperada, e no caso do RHAS, no qual se verificaram grandes magnitudes de deslocamentos em muitos nós presentes na malha inicial. É importante frisar que o cálculo dos esforços restauradores em todas as simulações presentes neste relatório foi feito através do método apresentado no Método B em 6.2.2. O Método A em 6.2.2 é igualmente eficiente para utilização com elementos de treliça, cabo e pórtico, apresentando a vantagem de estar formulado de maneira genérica, independente do tipo de elemento.

CAPÍTULO III – PRÉ-PROCESSADOR

III.1. Especificações Técnicas

Faz-se aqui necessária uma nova avaliação das especificações, uma vez que cada módulo do software é feito de maneira independente e, deve atender às suas necessidades específicas. Nesse módulo, as especificações a serem atendidas são:

- A criação de um ambiente gráfico, não necessariamente multiplataforma, com o qual seja possível a geração de um arquivo de entrada para o núcleo do ambiente.
- O software necessita ser "*user-friendly*", facilitando a entrada de dados e geração de malha, bem como a visualização da configuração inicial. Deve-se ressaltar que o objetivo é tornar simplificada e intuitiva a entrada de dados no programa central para o usuário ao qual a ferramenta se destina.
- Deve possuir funções de leitura e gravação de dados em XML, para que não seja necessária nenhuma intervenção do usuário no arquivo.
- Deve-se optar por uma linguagem de programação e bibliotecas gráficas que apresentem um compromisso entre velocidade e requisitos de *hardware* compatíveis e que possam ser utilizadas na maioria dos computadores pessoais atuais.

III.1.1. Linguagem de Programação

Com base nos requisitos anteriormente descritos, definiu-se que o sistema operacional alvo seria o *Microsoft Windows*®, por ser bastante difundido e estar instalado na grande maioria das empresas, laboratórios e residências, possibilitando uma maior abrangência do software no tangente aos usuários finais. Assim, a linguagem de programação escolhida foi o *C#* , componente da estrutura da *Microsoft*® denominada *.NET*, que é executável em qualquer sistema Windows e a biblioteca gráfica a ser utilizada o *OpenGL*®. As justificativas das escolhas feitas são detalhadas nos subitens a seguir.

Linguagem C#

Como foi feito anteriormente, levantaram-se todas as possibilidades de linguagens a serem utilizadas na confecção do pré-processador. De início descartaram-se todas as linguagens de baixo nível e sem orientação a objetos, já que nessa etapa busca-se a criação de diálogos com menus e recursos gráficos e tais linguagens quando permitem a utilização deste tipo de recurso tornam-se demasiadamente complicadas e não confiáveis.

Assim, partiu-se para a escolha da linguagem dentro das de alto-nível com orientação a objetos e possibilidade de criação de uma interface gráfica, chegando-se então às linguagens C++, *Managed* C++, C# e Java.

Considerou-se inicialmente que a linguagem escolhida deveria possuir um bom desempenho quando em execução, sendo essa característica primordial, uma vez que o tratamento gráfico demanda maiores recursos do equipamento e se a linguagem já comprometesse parte desse

recurso, o projeto ficaria inviabilizado. Assim descartou-se a linguagem Java por ser interpretada.

A escolha natural seria o C++ ou o *Managed* C++ e, por isso, foi alvo de algumas investigações. Primeiro concluiu-se que o *Managed* C++ oferecia uma grande vantagem: a possibilidade de uso dos Forms que substituem o Microsoft Fondation Classes (MFC). Como visto em Templeman, J., 2003 ([14])), o uso do MFC é desaconselhado em projetos novos, sendo só recomendado em casos que o programa precisa de pequenos ajustes, não compensando assim a sua conversão. Um ponto a ser ressaltado é que o *Managed* C++ é uma linguagem um pouco confusa, mesmo para pessoas acostumadas com o C++. Por isso, buscou-se conhecer uma opção na qual fosse possível usufruir de todo benefício do *Windows Forms*, o C#.

O C# pode ser descrito como uma linguagem muito semelhante ao C++ e ao Java, com as melhores coisas que podem ser encontradas em ambos. Possui uma estruturação parecida com Java em termos de classes , com o uso do *garbage collector,* e permite toda a encapsulação, generalidade, polimorfismo e possibilidade de se reescrever operadores encontradas no C++, além do uso dos *Forms*. Como desvantagens do C# tem-se que ela é parcialmente interpretada, sendo compilada para uma linguagem denominada *Microsoft Commom Runtime Language,* como pode ser visto em Sharp, J., 2003 ([15])) e também em Schildt, H., 1998 ([16]) e que possui certas limitações quanto à linguagem, se comparadas com o C++, como por exemplo, a proibição de reescrever o operador de atribuição de uma classe.

Considerando todas as vantagens e desvantagens oferecidas por cada linguagem, bem como o nível de conhecimento de cada uma, além de uma previsão de trabalho – tanto para o aprendizado da linguagem quanto para implementação de métodos – baseado na experiência e observação dos autores optou-se por usar o C#.

Biblioteca Gráfica OpenGL®

A biblioteca gráfica a ser usada foi decidida de uma maneira muito mais simples do que a linguagem de programação, já que a escolha se restringiu as duas mais conhecidas: DirectX da *Microsoft* e *Open Graphics Library (OpenGL®)* da *Silicon Graphics Internationa (SGI),* pois estas possuem maior literatura disponível e também são as mais utilizadas em desenvolvimento gráfico.

Passou-se por uma fase de estudo de ambas e concluiu-se que o uso do DirectX seria inviável dentro do prazo previsto para a execução do projeto, pois demandaria um tempo de aprendizado muito maior, já que este permite apenas a geração de gráficos por *Shaders* e por *Pipelines* que utilizam uma linguagem de baixo nível de acesso ao hardware gráfico. Por outro lado, a biblioteca *OpenGL®* possui implementação na linguagem C , e é apresentada por um grande número de livros e sites especializados no assunto (vide [17] a [20])), sendo o site NeHe Gamedev ([18]) o principal, que contém diversos tutoriais e é recomendado pela SGI.

O uso do *OpenGL®* inicialmente mostra-se conflitante com a escolha da linguagem de programação, uma vez que essa biblioteca não é escrita em C#. Tal problema foi resolvido de uma maneira muito simples. Com uma pesquisa no site oficial do *OpenGL®* ([17]) , percebeu-se que além dos arquivos *headers* do C, a implementação consistia em um conjunto de bibliotecas de vínculo dinâmico (em inglês DLL -Dynamic Link Library). Assim, se fosse possível importar as funções dessas DLL's para o C#, ou mesmo criar uma DLL própria, o problema seria resolvido.

Tentou-se inicialmente a criação da DLL baseando todo o raciocínio sobre o que foi encontrado em um site especializado em programação ([21]), o qual explicava como trabalhar

com *OpenGL*® e C# em uma mesma aplicação. Porém, tal solução mostrou-se demasiadamente trabalhosa, uma vez que necessitaria do aprendizado de três tipos de linguagens (C++, *Managed* C++ e C#).

Por isso, a opção adotada foi baseada na implementação do Framework TAO, *freeware* e *opensource* ([22]) . Gerou-se um conjunto de classes que consiste em importar as funções da biblioteca *OpenGL*® diretamente de suas DLL's, bem como algumas funções auxiliares e também uma implementação de um controle *OpenGL*® para ser utilizado nos *Forms*.

Uma vez estabelecido a linguagem de programação, a biblioteca e o modo como seria usado o *OpenGL*® no C#, partiu-se para a concepção do programa.

III.2. Estimativa Inicial da Configuração – Solução de um Cabo com n Trechos em Catenária

Um dos grandes objetivos do pré-processador é facilitar a entrada de dados para análise. Assim, deseja-se que o usuário possa delimitar um cabo de coordenadas extremas conhecidas, com trechos de comprimento e peso efetivo conhecidos e com base nisso possa gerar uma configuração similar à configuração inicial do cabo. Do ponto de vista computacional, isso facilita o cálculo pelo núcleo do programa, uma vez que a estimativa inicial da configuração dos cabos é baseada em soluções analíticas, que refletem a física real do problema e, do ponto de vista do usuário, permite uma visualização prévia da configuração antes da imposição de carregamento e condições de contorno.

Assim, criou-se um algoritmo que estima a configuração inicial de um cabo em catenária. O algoritmo foi generalizado de tal forma que dado um número qualquer de trechos com

comprimentos e pesos específicos conhecidos, bem como as posições extremas, fosse capaz de determinar os pontos intermediários e os parâmetros que definem a curva (dados através da solução analítica detalhada mais à frente). O procedimento para a criação do algoritmo será descrito nos sub-itens a seguir, da forma como o raciocínio foi construído.

Inicialmente determinam-se os dados de entrada e incógnitas para o caso de um cabo composto por um único trecho. Passa-se então para o caso com três trechos e com base neste generaliza-se o problema e assim tem-se o algoritmo desejado.

III.2.1. Solução Para uma Catenária Simples

Para facilitar, utilizar-se-á o mesmo raciocínio em todos os casos, sendo inicialmente listados os parâmetros de entrada, bem como as incógnitas. Porém, neste subitem serão retomadas as equações para um cabo em catenária entre dois pontos descritos no Capítulo II.

Como visto no referido capítulo, a solução da equação diferencial leva à equação (III.1).

$$y(x) = \frac{H}{\gamma}\cosh\left(\frac{\gamma}{H}x + A\right) + B \qquad \text{(III.1)}$$

Tal equação permite descrever a catenária, sendo dado qualquer valor da abscissa, a ordenada é imediatamente obtida. Observa-se que o sinal, que fica indeterminado na solução pode ser dado pelo sinal do γ, já que se o peso efetivo é negativo, a curva toma uma forma idêntica à positiva, porém simétrica em relação ao eixo das abscissas. Acresce-se a essa, a equação para cálculo do comprimento do cabo, que pode ser obtida facilmente, sendo tal demonstração feita a seguir.

Inicialmente, determina-se uma parametrização para a curva, sendo a escolhida a exibida em (III.2).

$$\Gamma(x) = \left(x, \frac{H}{\gamma} \cosh\left(\frac{\gamma}{H} x + A \right) + B \right) \tag{III.2}$$

E a partir disso, utiliza-se a formulação de cálculo de comprimento de uma curva no espaço, como na (III.3).

$$L = \int_{x_0}^{x_f} \left| \frac{d\Gamma}{dx} \right| dx \tag{III.3}$$

Assim, calcula-se o módulo da derivada da curva (III.4), obtém-se (III.5).

$$\frac{d\Gamma}{dx}(x) = \left(1, \sinh\left(\frac{\gamma}{H} x + A \right) \right) \tag{III.4}$$

$$\left| \frac{d\Gamma}{dx}(x) \right| = \sqrt{1 + \sinh^2\left(\frac{\gamma}{H} x + A \right)} \tag{III.5}$$

Lembrando que $1 + \sinh^2(\alpha) = \cosh^2(\alpha)$ tem-se o módulo dado por (III.6).

$$\left| \frac{d\Gamma}{dx}(x) \right| = \cosh\left(\frac{\gamma}{H} x + A \right) \tag{III.6}$$

E com isso tem-se uma fórmula bem simplificada para o comprimento (III.7).

$$L = \int_{x_0}^{x_f} \cosh\left(\frac{\gamma}{H} x + A \right) dx = \left[\frac{H}{\gamma} \sinh\left(\frac{\gamma}{H} x + A \right) \right]_{x_0}^{xf} \tag{III.7}$$

E finalmente, a expressão final para o comprimento (III.8).

$$L = \frac{H}{\gamma} \sinh\left(\frac{\gamma}{H} x_f + A \right) - \frac{H}{\gamma} \sinh\left(\frac{\gamma}{H} x_0 + A \right) \tag{III.8}$$

Assim, para cada cabo é possível escrever três equações, sendo duas delas a partir das coordenadas extremas do cabo e, uma, a partir do comprimento do cabo.

Uma vez que foram descritas as equações da catenária, pode-se passar para a listagem os parâmetros de entrada e incógnitas, feita na Tabela 11 e na Tabela 12 Para uniformizar a nomenclatura, mesmo com um trecho, o cabo receberá um índice subscrito 1.

Tabela 11 – Parâmetros de Entrada – Catenária Simples

Parâmetro	***Símbolo do parâmetro***
Ponto inicial – Coordenada X	x_0
Ponto inicial – Coordenada Y	y_0
Peso específico do trecho	γ_1
Comprimento do trecho	L_1
Ponto final – Coordenada X	x_f
Ponto final – Coordenada Y	y_f

Tabela 12 – Incógnitas– Catenária Simples

Incógnita	***Símbolo da incógnita***
Força Horizontal	H
Variável de Integração	A_1
Variável de Integração	B_1

Um comentário a ser feito é que a força horizontal não possui índice porque a mesma mantém-se constante no cabo inteiro, independentemente do número de trechos, o que pode

ser demonstrado fazendo o equilíbrio de forças na direção horizontal em todos os pontos do cabo.

De posse das equações que relacionam os parâmetros e variáveis é possível a construção de um sistema matricial não-linear com base nas equações do sistema (III.9).

$$\begin{cases} f_1(x_0, y_0, \gamma_1, L_1, x_f, y_f, H, A_1, B_1) = 0 \\ f_2(x_0, y_0, \gamma_1, L_1, x_f, y_f, H, A_1, B_1) = 0 \\ f_3(x_0, y_0, \gamma_1, L_1, x_f, y_f, H, A_1, B_1) = 0 \end{cases} \tag{III.9}$$

Para a resolução do sistema, foi utilizado o método de Newton para sistemas não-lineares, que é descrito no Apêndice E.

Deve-se ressaltar que a ordem das equações na matriz interfere na resolução do problema, levando inclusive à não-convergência do método no caso de uma ordenação ruim, possivelmente causando condicionamento matricial ruim. Assim, a ordenação feita para o caso de existir apenas um trecho é a seguinte: A_1, H, B_1, pois dessa forma podem ser evitados valores nulos na diagonal principal, que eventualmente seriam problemáticos para o método de resolução utilizado. Consequentemente as equações ficam ordenadas na seguinte seqüência: equação do extremo inicial (III.10), equação do comprimento (III.11) e equação do extremo final (III.12).

$$y_0 = \frac{H}{\gamma_1}\cosh\left(\frac{\gamma_1}{H}x_0 + A_1\right) + B_1 \tag{III.10}$$

$$L_1 = \frac{H}{\gamma_1}\sinh\left(\frac{\gamma_1}{H}x_f + A_1\right) - \frac{H}{\gamma_1}\sinh\left(\frac{\gamma_1}{H}x_0 + A_1\right) \tag{III.11}$$

$$y_f = \frac{H}{\gamma_1}\cosh\left(\frac{\gamma_1}{H}x_f + A_1\right) + B_1 \tag{III.12}$$

Colocando na forma Matricial, obtêm-se os vetores **x** das incógnitas (III.13), **F** das funções a serem zeradas (III.14) e a matriz **J** do jacobiano das funções (III.15).

$$\mathbf{x} = \begin{Bmatrix} A_1 \\ H \\ B_1 \end{Bmatrix} \tag{III.13}$$

$$\mathbf{F} = \begin{Bmatrix} \dfrac{H}{\gamma_1}\cosh\left(\dfrac{\gamma_1}{H}x_0 + A_1\right) + B_1 - y_0 \\ \dfrac{H}{\gamma_1}\sinh\left(\dfrac{\gamma_1}{H}x_f + A_1\right) - \dfrac{H}{\gamma_1}\sinh\left(\dfrac{\gamma_1}{H}x_0 + A_1\right) - L_1 \\ \dfrac{H}{\gamma_1}\cosh\left(\dfrac{\gamma_1}{H}x_f + A_1\right) + B_1 - y_f \end{Bmatrix} \tag{III.14}$$

$$\mathbf{J} = \begin{bmatrix} \dfrac{H}{\gamma_1}\sinh\left(\dfrac{\gamma_1}{H}x_0 + A_1\right) & \dfrac{1}{\gamma_1}\cosh\left(\dfrac{\gamma_1}{H}x_0 + A_1\right) - \dfrac{x_0}{\gamma_1}\sinh\left(\dfrac{\gamma_1}{H}x_0 + A_1\right) & 1 \\ \dfrac{H}{\gamma_1}\left[\cosh\left(\dfrac{\gamma_1}{H}x_f + A_1\right) - \cosh\left(\dfrac{\gamma_1}{H}x_0 + A_1\right)\right] & \dfrac{L_1}{H} - \dfrac{1}{H}\left[x_f\cosh\left(\dfrac{\gamma_1}{H}x_f + A_1\right) - x_0\cosh\left(\dfrac{\gamma_1}{H}x_0 + A_1\right)\right] & 0 \\ \dfrac{H}{\gamma_1}\sinh\left(\dfrac{\gamma_1}{H}x_f + A_1\right) & \dfrac{1}{\gamma_1}\cosh\left(\dfrac{\gamma_1}{H}x_f + A_1\right) - \dfrac{x_0}{\gamma_1}\sinh\left(\dfrac{\gamma_1}{H}x_f + A_1\right) & 1 \end{bmatrix} \tag{III.15}$$

III.2.2. Solução Para uma Catenária de Três Trechos

Para três trechos, o raciocínio foi análogo, porém resultando em um sistema de ordem nove. Inicialmente serão listados os parâmetros do cabo (Tabela 13) e as incógnitas associadas ao problema (Tabela 14), como feito anteriormente.

Tabela 13 – Parâmetros de Entrada – Catenária com Três Trechos

Parâmetro	***Símbolo do parâmetro***
Ponto inicial – Coordenada X	x_0
Ponto inicial – Coordenada Y	y_0
Peso específico do trecho um	γ_1
Comprimento do trecho um	L_1
Peso específico do trecho dois	γ_2
Comprimento do trecho dois	L_2
Peso específico do trecho três	γ_3
Comprimento do trecho três	L_3
Ponto final – Coordenada X	x_f
Ponto final – Coordenada Y	y_f

Tabela 14 - Incógnitas– Catenária com Três Trechos

Incógnita	***Símbolo da incógnita***
Força Horizontal	H
Variável de Integração	A_1
Variável de Integração	B_1
Ponto Intermediário entre trechos um e dois	x_1
Variável de Integração	A_2
Variável de Integração	B_2
Ponto Intermediário entre trechos dois e três	x_2
Variável de Integração	A_3
Variável de Integração	B_3

Nota-se que aparecem como incógnitas os pontos intermediários Ressalta-se que os valores de *y* dos pontos intermediários não são incógnitas, pois, uma vez determinada as coordenadas *x* intermediárias entre os trechos de cabo, os valores de *y* correspondentes podem ser obtidos diretamente pela solução analítica.

Nesse caso, a ordenação do vetor **x** escolhida é a seguinte: A_1, x_1, B_1, A_2, x_2, B_2, A_3, H, B_3, sendo essa tal que evita um mal-comportamento numérico já explicado para o caso de apenas um trecho de cabo. A seguir são exibidas as equações que compõem o vetor **F** – equações (III.16) a (III.24) – e que servirão para o cálculo da matriz Jacobiana.

$$y_0 = \frac{H}{\gamma_1}\cosh\left(\frac{\gamma_1}{H}x_0 + A_1\right) + B_1 \tag{III.16}$$

$$L_1 = \frac{H}{\gamma_1}\sinh\left(\frac{\gamma_1}{H}x_1 + A_1\right) - \frac{H}{\gamma_1}\sinh\left(\frac{\gamma_1}{H}x_0 + A_1\right) \tag{III.17}$$

$$\frac{H}{\gamma_1}\cosh\left(\frac{\gamma_1}{H}x_1 + A_1\right) + B_1 = \frac{H}{\gamma_2}\cosh\left(\frac{\gamma_2}{H}x_1 + A_2\right) + B_2 \tag{III.18}$$

$$\sinh\left(\frac{\gamma_1}{H}x_1 + A_1\right) = \sinh\left(\frac{\gamma_2}{H}x_1 + A_2\right) \tag{III.19}$$

$$L_2 = \frac{H}{\gamma_2}\sinh\left(\frac{\gamma_2}{H}x_2 + A_2\right) - \frac{H}{\gamma_2}\sinh\left(\frac{\gamma_2}{H}x_1 + A_2\right) \tag{III.20}$$

$$\frac{H}{\gamma_2}\cosh\left(\frac{\gamma_2}{H}x_2 + A_2\right) + B_2 = \frac{H}{\gamma_3}\cosh\left(\frac{\gamma_3}{H}x_2 + A_3\right) + B_3 \tag{III.21}$$

$$\sinh\left(\frac{\gamma_2}{H}x_2 + A_2\right) = \sinh\left(\frac{\gamma_3}{H}x_2 + A_3\right) \tag{III.22}$$

$$L_3 = \frac{H}{\gamma_3}\sinh\left(\frac{\gamma_3}{H}x_f + A_3\right) - \frac{H}{\gamma_3}\sinh\left(\frac{\gamma_3}{H}x_2 + A_3\right) \tag{III.23}$$

$$y_f = \frac{H}{\gamma_3}\cosh\left(\frac{\gamma_3}{H}x_f + A_3\right) + B_3 \tag{III.24}$$

Observa-se que (III.18) e (III.19) são equações de compatibilidade entre os trechos um e dois, impondo respectivamente a mesma posição e ângulo dos cabo na junção. Analogamente, (III.21) e (III.22) impõem essas condições para a junção dos trechos dois e três.

III.2.3. Generalização para Catenária de n Trechos

Para a generalização do código, alguns pontos devem ser considerados. Primeiro, observa-se que a inclusão de cada trecho aumenta em três o número de incógnitas do problema, mas ao mesmo tempo fornece mais três equações. Segundo, percebe-se um padrão nesse conjunto de equações adicionadas, composto de duas equações de compatibilidade de posição e ângulo e uma equação referente ao comprimento do trecho. Outro ponto a ser observado é que a ordenação das variáveis que foi feita permite a inclusão de um bloco intermediário, sem causar mal condicionamento numérico.

A seguir, são listadas as incógnitas (Tabela 15) e os parâmetros do cabo (Tabela 16) para o caso genérico de *n* trechos.

Tabela 15 – Parâmetros de Entrada – Catenária com *n* Trechos

Parâmetro	***Símbolo do parâmetro***
Ponto inicial – Coordenada X	x_0
Ponto inicial – Coordenada Y	y_0
Peso específico do trecho i	γ_i
Comprimento do trecho i	L_i
Ponto final – Coordenada X	x_f
Ponto final – Coordenada Y	y_f

Tabela 16 – Incógnitas– Catenária com *n* Trechos

Incógnita	***Símbolo da incógnita***
Força Horizontal	H
Variável de Integração	A_1
Variável de Integração	B_1
Ponto Intermediário entre trechos i e i+1	x_i
Variável de Integração	A_i
Variável de Integração	B_i
Variável de Integração	A_n
Variável de Integração	B_n

Onde i varia de um ao n-1, onde n é o número de trechos. A ordenação das variáveis fica: A_1, x_1, B_1, A_i, x_i, B_i,..., A_n, H, B_n. Percebe-se que as equações, quando não envolvem apenas variáveis do mesmo trecho, envolvem no máximo variáveis do trecho seguinte, gerando

alguns “blocos” na matriz. A única incógnita que aparece em todas as equações é a força horizontal. Assim, determinou-se que essa sempre fica na penúltima posição do vetor, independentemente do número de trechos.

Com todas essas observações, foi possível a criação do algoritmo generalizado para *n* trechos de cabo em catenária para um cabo qualquer usando o método de Newton aplicado à resolução de sistemas não-lineares.

III.3. Descrição do módulo de entrada de dados

Esse módulo constitui a interface entre o usuário e o programa de resolução de cabos em configuração estática, sendo de extrema importância para a geração de casos com diversos elementos, nós, carregamentos e com corrente.

A Figura 27 mostra a janela principal do programa, a qual é descrita com mais detalhes logo adiante.

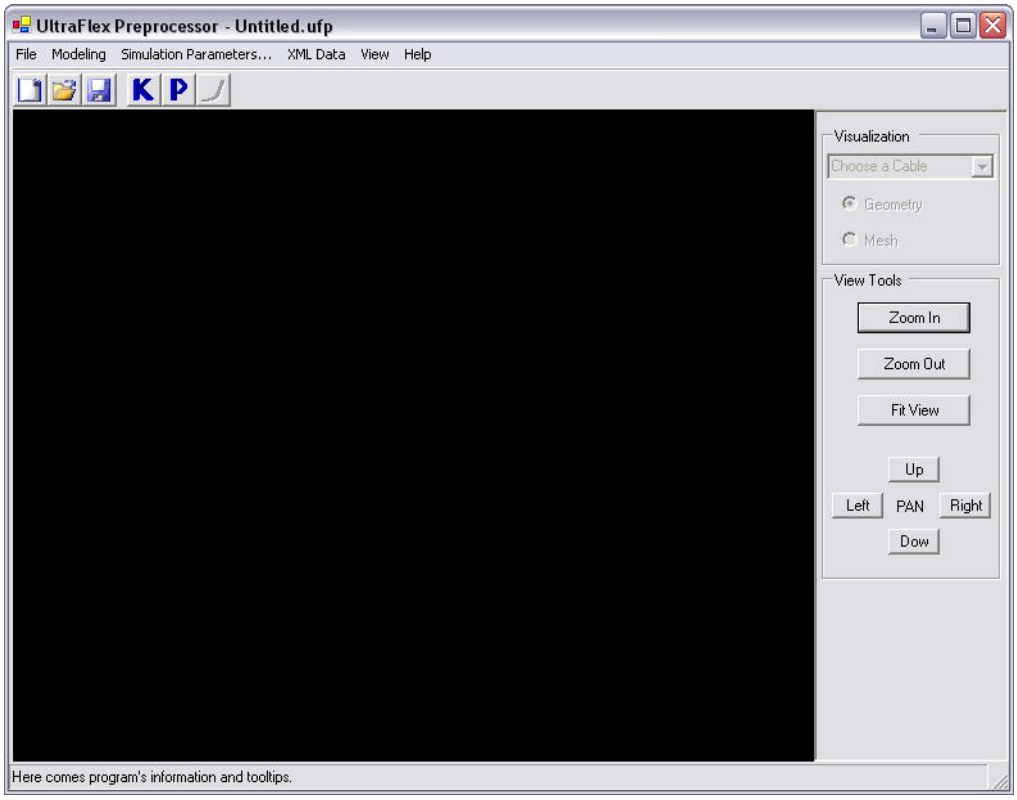


Figura 27 – Janela principal do programa

Como pode ser visto, a janela é constituída de uma barra de títulos, na qual é exibido o nome do programa – UltraFlex Preprocessor –, bem como o arquivo aberto. No caso da Figura 27, o arquivo aberto é o inicial: um caso sem nome e sem nenhum dado. No inferior da mesma observa-se uma barra de status, na qual são exibidas mensagens ao usuário, informando-o sobre uma ação ou exibindo uma dica sobre o objeto apontado. Entendem-se por objetos todos os botões e menus visíveis nessa janela. Também pode ser visto uma barra de menus que será descrita com maiores detalhes nos próximos itens.

Destacam-se três partes fundamentais neste pré-processador: a barra de ferramentas, a barra de visualização e a área de desenho.

III.3.1. Barra de ferramentas

A barra de ferramentas é exibida em detalhe na Figura 28.

Figura 28 – Barra de ferramentas

O primeiro botão (*New*) permite a criação de um novo projeto. O segundo botão (*Open*) permite a abertura de um arquivo e o terceiro botão (*Save*), permite que o arquivo atual seja salvo. Esses comandos serão descritos com mais detalhes no item III.4.1.

O segundo grupo de botões são atalhos para as três primeiras janelas que permitem a modelagem do problema. São na ordem: *Keypoints*, *Properties* e *Cables*, que serão descritas com mais detalhes no item III.4.2.

III.3.2. Área de desenho

Nesse espaço reservado (área preta na Figura 27) é que serão desenhadas as estimativas iniciais do cabo, bem como as suas malhas, permitindo ao usuário uma maior percepção do problema (e também de uma possível configuração, já que a estimativa inicial é feita por composição de diversos trechos de catenária, como descrito no item III.2). Essa região nada mais é que um controle implementado utilizando-se algumas bibliotecas *OpenGL*®.

III.3.3. Barra de Visualização

A barra de visualização permite um controle da área desenhada do cabo o qual está selecionado. Essa barra, como pode ser visto na Figura 29, permite primeiramente a escolha do cabo a ser visualizado, seguido do tipo de visualização: geometria ou malha e possui também botões que permitem ao usuário visualizar as partes desejadas do cabo. Todos os comandos do grupo *View Tools* serão descritos com mais detalhes no menu View (item III.4.5).

Figura 29 – Barra de ferramentas de visualização

Os comandos do grupo *Visualization* estão presentes apenas nesta barra e, portanto serão descritos aqui. Permite-se a escolha do cabo a ser visualizado ou o conjunto de todos os cabos e *keypoints* atualmente inseridos.

A seguir permite-se a escolha entre malha e geometria. A Figura 30 mostra o caso em que a geometria é exibida, enquanto a Figura 31 mostra o caso em que é exibida a malha.

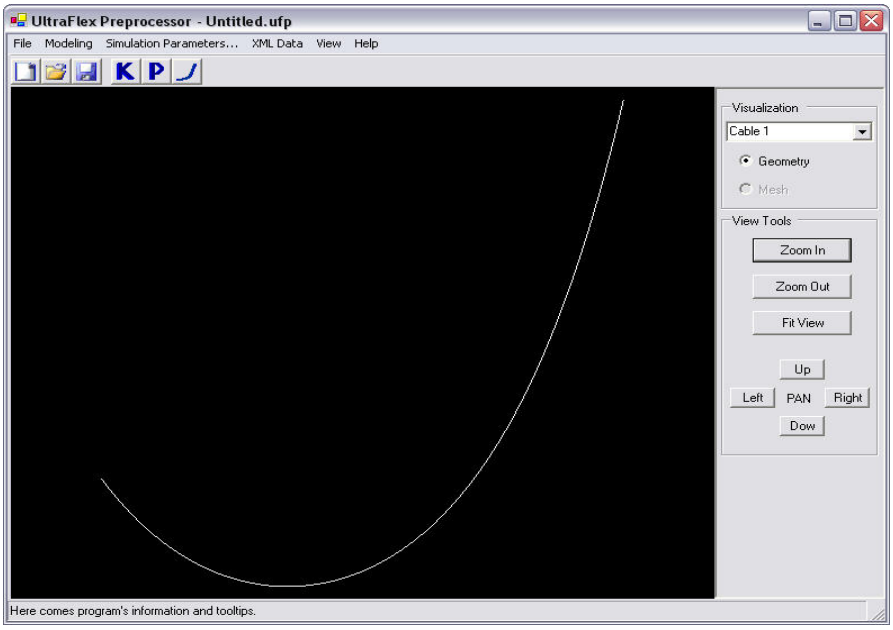


Figura 30 – Janela principal exibindo geometria

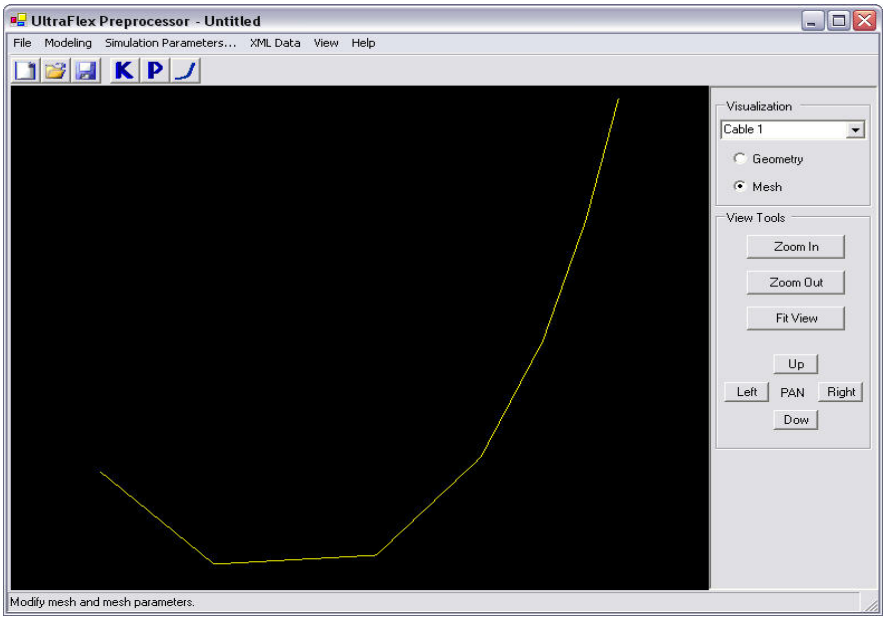


Figura 31 – Janela principal exibindo malha

III.4. Descrição dos menus e das janelas componentes do módulo

O menu principal foi construído de tal forma que permite ao usuário uma maior facilidade na inclusão de dados e para tal, foi feito de forma intuitiva. Procurou-se seguir uma ordem lógica comum na construção de um modelo de cabo a ser simulado. A seguir são descritos com mais detalhes todos os itens nos menus, exibindo as janelas para as quais eles permitem acesso, bem como uma descrição das mesmas.

III.4.1. Menu File

O menu *File* é mostrado na Figura 32. Possui os itens *New*, *Open*, Save, *Save As*, *Close* e *Exit* que são descritos a seguir.

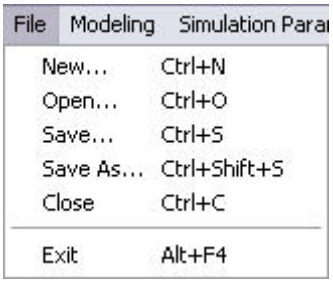


Figura 32 – Menu File

- ***Comando New:***

O comando New permite a criação de um caso vazio e sem título.

- ***Comando Open:***

O comando Open permite a abertura de um arquivo do tipo "*ufp*" que contém dados anteriormente salvos do programa. A Figura 33 mostra a tela de abertura de arquivo quando o botão é pressionado.

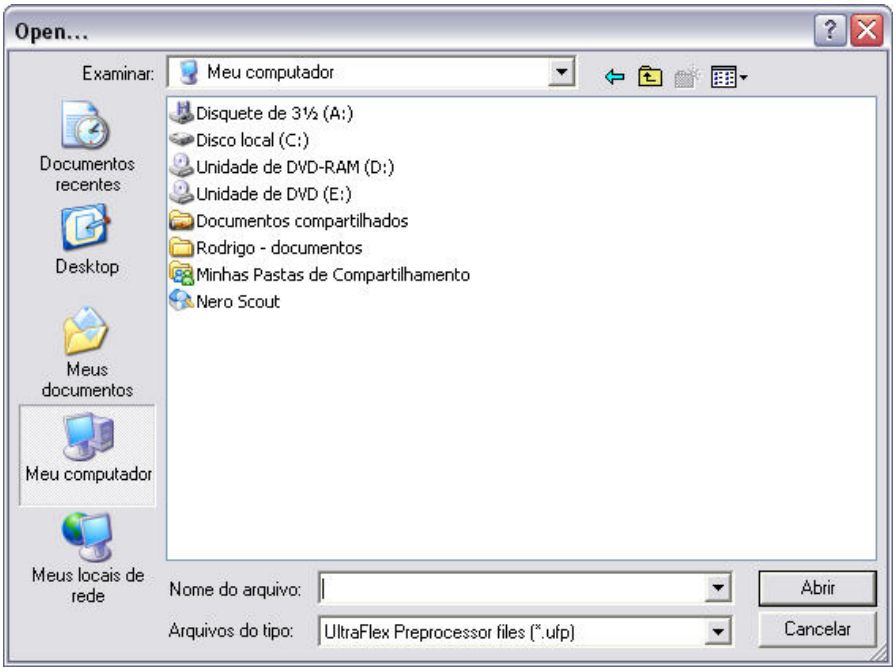


Figura 33 – Janela de abertura de arquivo

- ***Comando Save:***

O último botão do primeiro grupo (*Save*) permite que o arquivo atual seja salvo. Caso o nome não tenha sido especificado anteriormente, abre a janela que permite que o mesmo seja salvo (Figura 34).

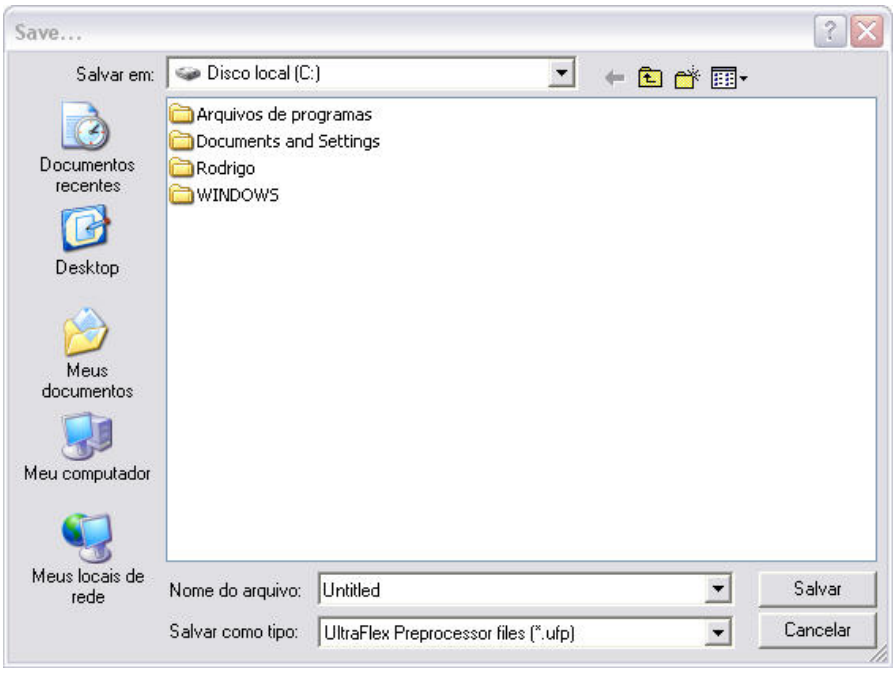


Figura 34 – Janela que permite que o arquivo seja salvo com o nome desejado

- ***Comando Save As:***

Esse comando permite salvar um arquivo já salvo com um outro nome. Abre uma janela semelhante à exibida na Figura 34.

- ***Comando Close:***

Encerra o arquivo atual, esperando a abertura de um novo arquivo ou um arquivo anteriormente salvo.

- ***Comando Exit:***

Encerra o programa *UltraFlex Preprocessor*.

III.4.2. Menu Modeling

O menu *Modeling* é mostrado na Figura 35. Esse é o principal do programa, uma vez que permite o acesso às janelas nas quais são inseridos os dados referentes ao modelo a ser estudado. Possui os seguintes comandos: Environment, *Keypoints*, *Properties*, *Cables*, *Mesh*, *Constraints* e *Loads*, sendo que dentro do menu *Loads* encontram-se os subitens *Nodal Loads* e *Current*.

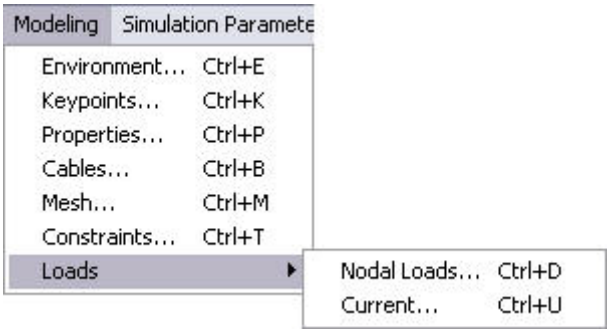


Figura 35 – Menu Modeling

- ***Comando Environment:***

O comando *Environment* abre a janela que permite a configuração do ambiente no qual o modelo será simulado. Essa janela é exibida na Figura 36.

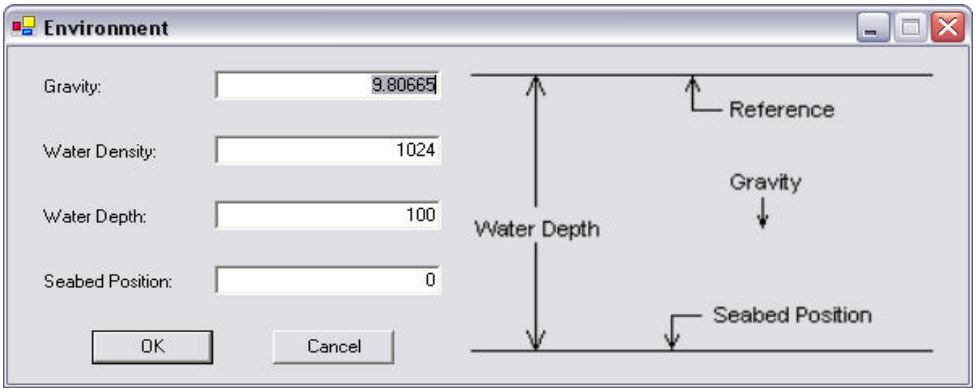


Figura 36 – Janela de configuração do ambiente

Nesse diálogo é possível inserir os valores da aceleração da gravidade local, a densidade da água, bem como a lâmina d'água e a posição do solo.

- ***Comando Keypoints:***

Keypoints são pontos no espaço que servem para auxiliar o projeto, indicando locais nos quais serão gerados futuros nós, sendo de fundamental importância na definição do problema. O comando *Keypoints* faz com que a janela de inserção de *keypoints* seja aberta. A Figura 37 mostra essa janela.

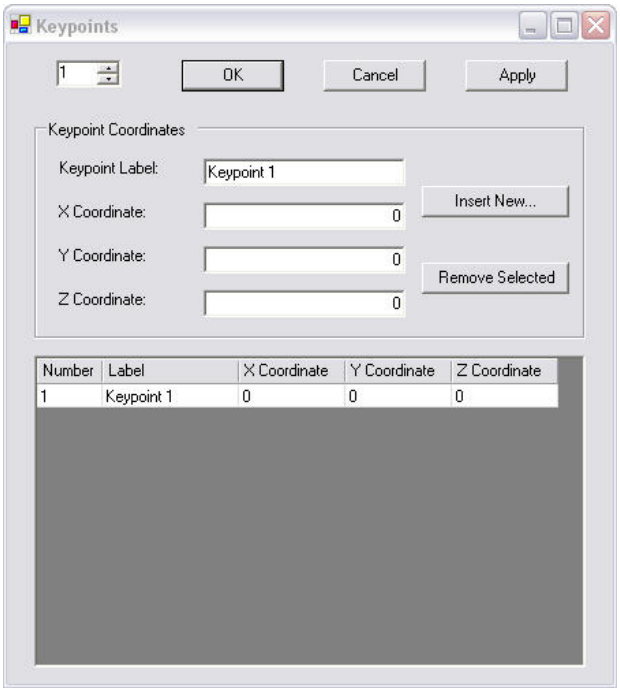


Figura 37 – Janela de inserção de keypoints

Nessa janela podem ser adicionados ou excluídos os *keypoints* selecionados através do número, exibido no canto esquerdo superior do diálogo. A inclusão e remoção são feitas clicando-se nos botões *Insert New...* e *Remove Selected* respectivamente. Para modificar tanto o rótulo e as coordenadas desses *keypoints*, basta utilizar os campos no centro da tela. A tabela da parte inferior permite ver quais *keypoints* já existem, facilitando assim a compreensão do usuário.

Para que qualquer atualização seja feita, basta pressionar-se o botão *Apply*. Caso o usuário clique em *OK*, o diálogo é encerrado e as alterações são salvas. Caso o usuário clique em *Cancel*, as alterações feitas são perdidas, sendo que se o usuário clicou em *Apply* em algum momento na utilização deste diálogo, apenas as alterações posteriores são perdidas.

- ***Comando Properties:***

O comando *Properties* faz com que a janela de inserção de propriedades seja aberta. A Figura 38 mostra essa janela.

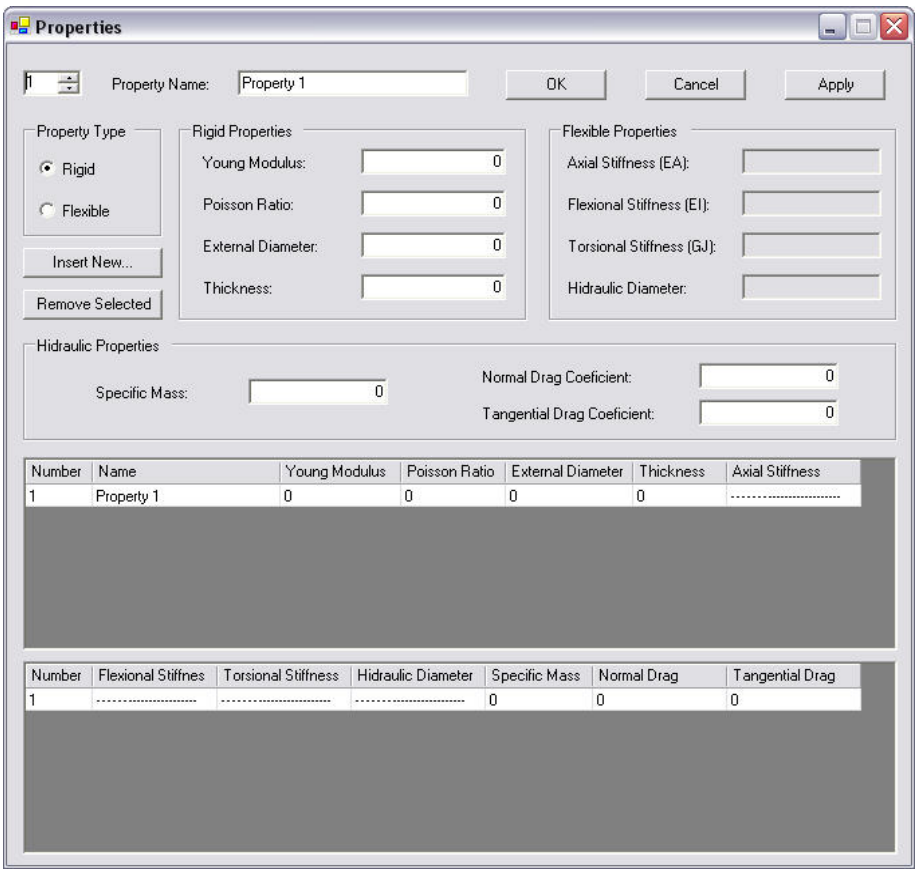


Figura 38 – Janela de inserção de propriedades

Nessa janela podem ser adicionados ou excluídos dados de propriedades relativas à cabos em geral, independentemente do seu tipo. Analogamente ao descrito anteriormente, nesse diálogo altera-se as propriedades do *Property* selecionados, sendo tal seleção feita através do número, exibido no canto esquerdo superior do diálogo. A inclusão e remoção são feitas clicando-se nos botões *Insert New...* e *Remove Selected*, como também foi descrito.

É possível uma escolha entre propriedades de cabos rígidos e flexíveis, sendo que as seguintes propriedades são comum a ambas:

- Um nome que o identifique;
- Massa por unidade de comprimento;
- Propriedades hidrodinâmicas: coeficiente de arrasto nas direções normal e tangencial.

Quando se escolhe o tipo do cabo como flexível, os dados a serem fornecidos são os módulos equivalentes (axial, flexão e torção) e também o diâmetro hidráulico. Quando o tipo é rígido, os dados a serem fornecidos são o módulo de elasticidade, coeficiente de Poisson, diâmetro e espessura do tubo.

Existe na parte inferior da janela uma tabela que permite ao usuário verificar as propriedades até o momento inseridas. Da mesma maneira que no diálogo dos *Keypoints*, para que qualquer atualização seja feita, basta pressionar-se o botão *Apply*. Os botões OK e *Cancel* possuem a mesma funcionalidade descrita anteriormente.

- ***Comando Cables:***

O comando *Cables* faz com que a janela que permite a criação do cabo seja exibida. O referido diálogo é exibido na Figura 39. Um ponto a ser considerado é que esse diálogo somente está disponível caso existam pelo menos dois *keypoints* e uma propriedade definida.

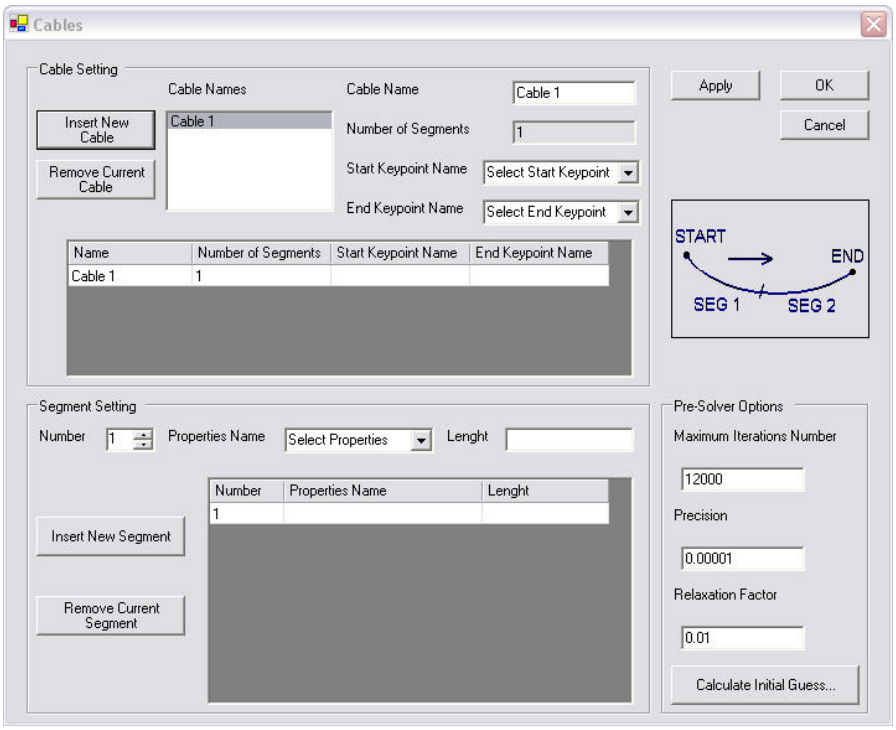


Figura 39 – Diálogo de inserção de cabos

O comando *Insert New Cable* insere um novo cabo na análise. É possível inserir quantos cabos forem desejados pelo usuário. Para cada um inserido é necessário indicar algumas características: nome do cabo, nome (rótulo) do *keypoint* de início, e do *keypoint* do fim da extensão do cabo. Essas informações de todos os cabos ficam armazenadas em uma tabela localizada abaixo de uma lista de nomes dos cabos previamente inseridos.

O usuário pode, através dessa lista, selecionar ainda os segmentos que existirão no interior de cada cabo. Assim, uma vez feita cada seleção, serão exibidos na parte inferior do diálogo os segmentos previamente existentes no cabo selecionado. A inserção de um segmento inclui a entrada de dois dados: o comprimento e a propriedade. As propriedades disponíveis são exibidas em um *combo box* acima da tabela de dados dos segmentos.

Após a entrada dos dados do cabo e, de seus segmentos, o diálogo exibe o número de segmentos inseridos, junto às propriedades do cabo.

A partir desses dados, e utilizando o método apresentado em III.2, é possível a partir das soluções analíticas para cada segmento, determinar uma estimativa inicial da malha a ser utilizada. A geração da malha ocorre quando o usuário aciona o comando *Calculate Initial Guess*. Como opções para esse pré-solver, estão o número de iterações máximo do método de Newton Raphson, o fator de relaxação do método e, a precisão requerida para o cálculo. Alguns valores padrões para esses dados já estão com valores padrões no diálogo. A resposta do pré-solver é exibida através de duas mensagens possíveis. A Figura 40 exibe a mensagem de quando não ocorre convergência no método, e a Figura 41 exibe a mensagem de convergência obtida com sucesso.

Os comandos *OK*, *Apply* e *Cancel* são equivalentes aos dos diálogos *Properties* e *Keypoints*, já apresentados.

Figura 40 – Erro na solução da estimativa inicial

Figura 41 – Sucesso na solução da estimativa inicial

- ***Comando Mesh:***

O comando *Mesh* exibe o diálogo que permite a inserção e modificação de malhas. Esse diálogo, exibido na Figura 42, só está disponível quando pelo menos um cabo está criado.

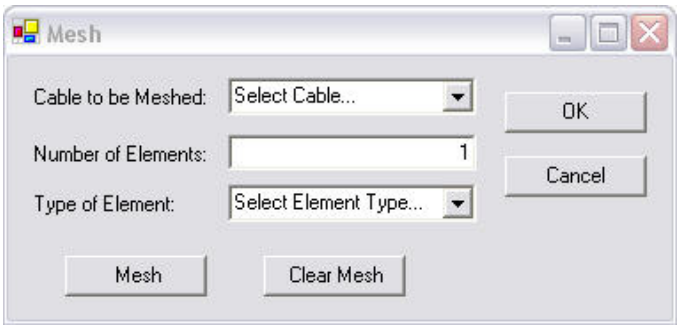


Figura 42 – Diálogo de geração de malha

Primeiramente, deve-se selecionar o cabo para qual deve ser gerada a malha, bem como o número de elementos e tipo de elemento desejado. Para gerar a malha automaticamente, clica-se no botão *Mesh*. No caso do usuário não colocar o cabo ao qual será aplicada a malha, é

exibida uma mensagem de erro (Figura 43). Também é exibida uma mensagem de erro caso não exista um tipo de elemento selecionado (Figura 44).

Figura 43 – Erro quando não há cabo selecionado

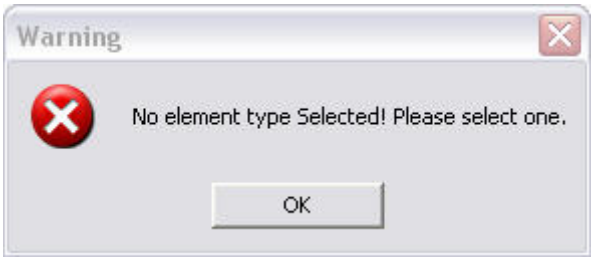


Figura 44 – Erro quando não há elemento selecionado

Caso este cabo possua uma malha anterior, uma mensagem de aviso será exibida (Figura 45).

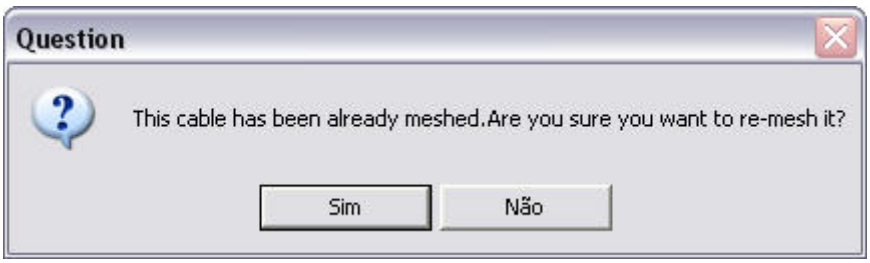


Figura 45 – Mensagem de aviso de malha pré-existente

Se o usuário desejar excluir a malha do cabo selecionado, basta que ele clique em *Clear Mesh*. Nesse caso, também será exibido um aviso alertando-o que está prestes a excluir a malha atual, como pode ser visto na Figura 46. Para que as alterações sejam feitas, clica-se em *OK*. Clicando-se em *Cancel*, as alterações não são salvas nas listas globais de nós e elementos.

Figura 46 – Mensagem de aviso de exclusão de malha

A seguir, na Figura 47, na Figura 48 e na Figura 49 são exibidos exemplos de malhas gerados com 12, 25 e 100 elementos, sendo que um exemplo de malha menos refinada – com 6 elementos – foi exibida anteriormente, na Figura 31.

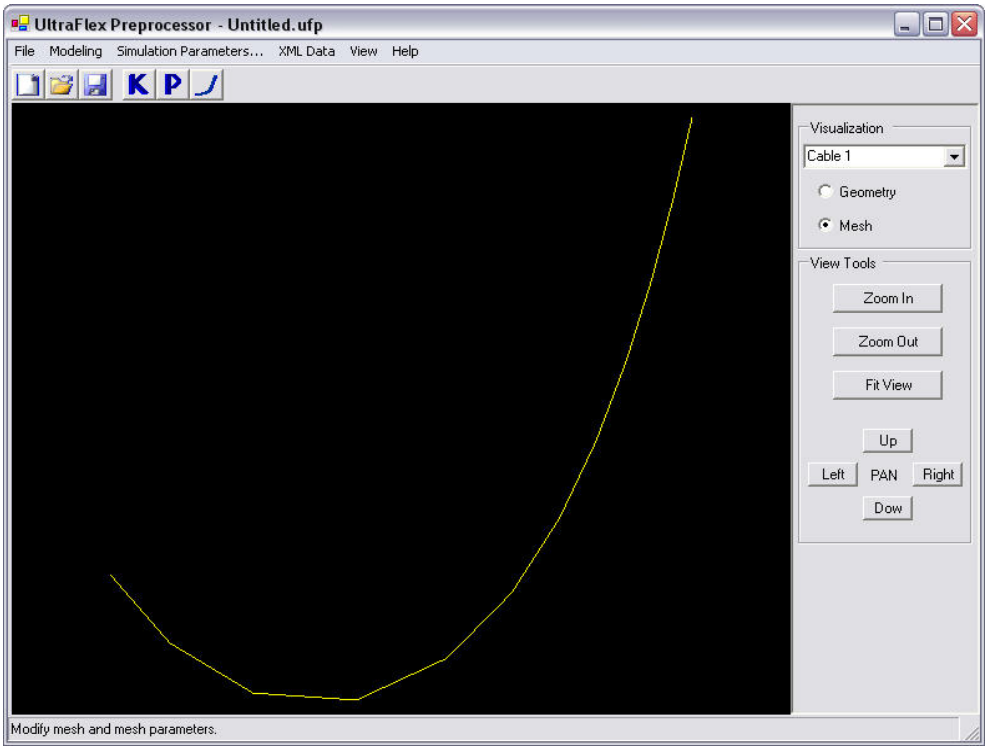


Figura 47 – Malha com 12 elementos

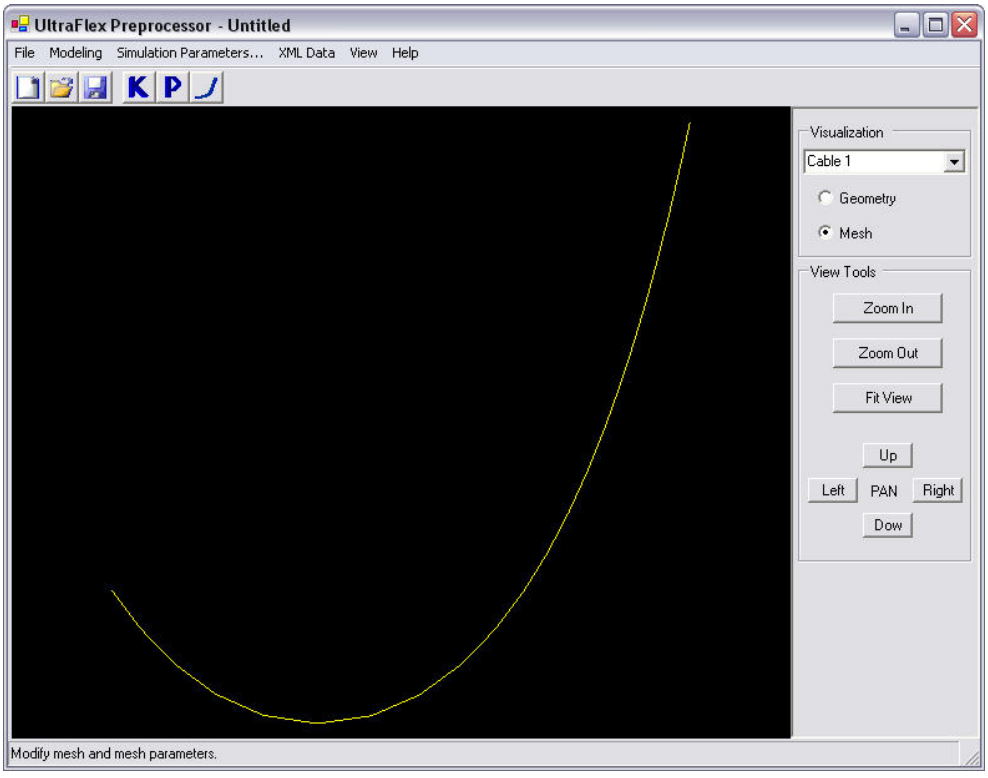


Figura 48 – Malha com 25 elementos

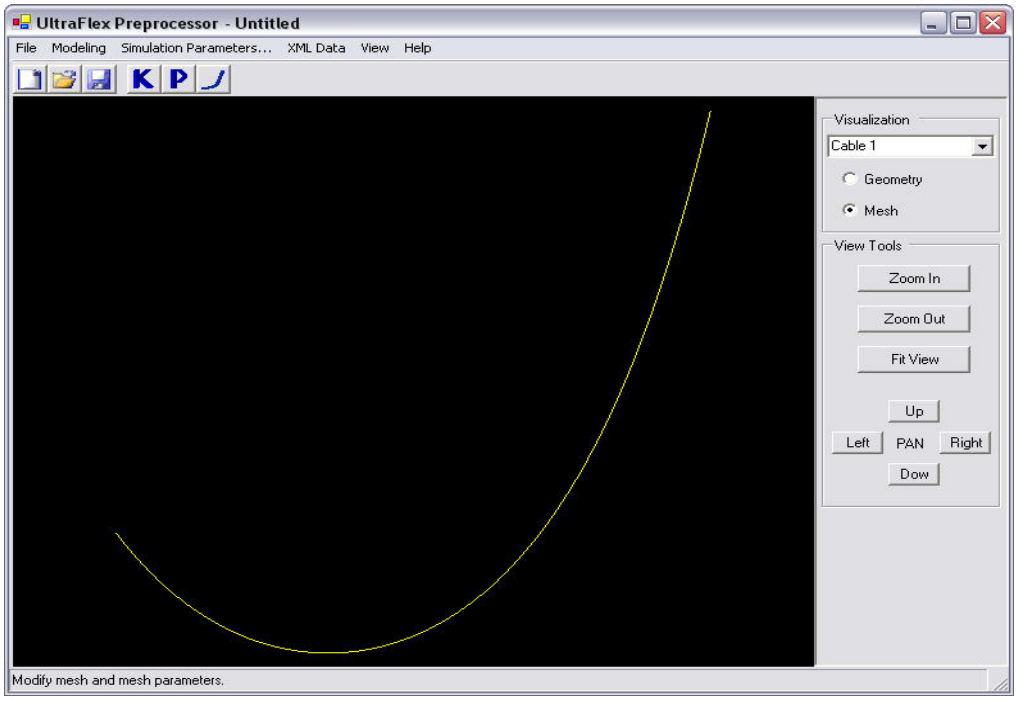


Figura 49 – Malha com 100 elementos

- ***Comando Constraints:***

O comando *Constraints* exibe o diálogo que permite a inserção e modificação de restrições nodais. Esse diálogo, exibido na Figura 50, só está disponível quando pelo menos um cabo possui malha.

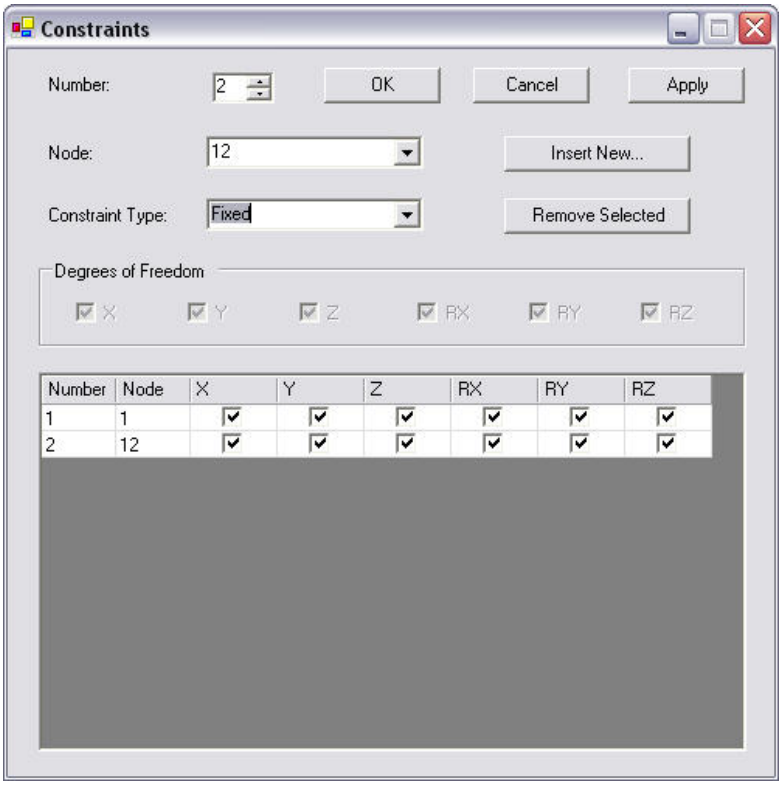


Figura 50 – Diálogo de inserção de restrições

Nesse diálogo selecionam-se qual o número da restrição a ser modificada, da mesma forma que nos diálogos *keypoints* e *properties*, permitindo a inserção e remoção dos mesmos através dos botões *Insert New...* e *Remove Selected*, respectivamente. Com uma restrição selecionada, é possível escolher-se o nó no qual essa agirá e também o tipo de restrição dentre uma lista: *fixed*, que não permite nenhum tipo de movimento nodal; *joint*, que permite apenas rotações; e *user defined*, que permite que o usuário escolha quais graus de liberdade serão presos. A escolha desses graus é feita através do quadro abaixo do menu de seleção do tipo da restrição, que se torna editável quando da escolha de *user defined*.

Os restantes dos botões funcionam também da mesma forma que nos diálogos *keypoints* e *properties*.

- ***Comando Loads – Nodal Loads:***

O comando *Nodal Loads* do submenu *Loads* exibe o diálogo que permite a inserção e modificação de carregamentos nodais. Esse diálogo, exibido na Figura 51, como no caso do diálogo de restrições, só está disponível quando pelo menos um cabo possui malha.

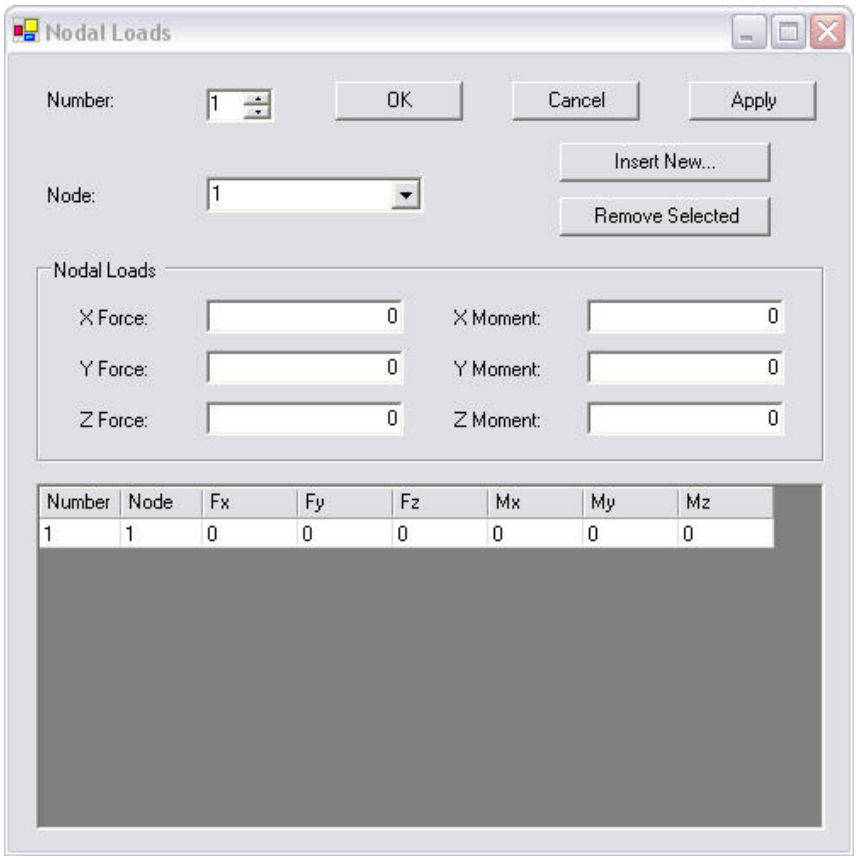


Figura 51 – Diálogo de carregamentos nodais

Nesse diálogo seleciona-se qual o número do carregamento nodal a ser modificado, da mesma forma que nos diálogos *keypoints* e *properties*, permitindo a inserção e remoção dos mesmos através dos botões *Insert New...* e *Remove Selected*, respectivamente. Com um carregamento selecionado, é possível escolher-se o nó no qual essa agirá, além de permitir a inserção das projeções das forças e momentos nos três eixos, gerando assim um conjunto de seis esforços.

Na parte inferior do diálogo é exibida uma tabela, que permite visualizar todos os carregamentos até agora inseridos. O restante dos botões funciona, também, da mesma forma do que nos diálogos *keypoints* e *properties*.

- ***Comando Loads – Current:***

O comando *Current* do submenu *Loads* exibe o diálogo que permite a inserção e modificação de uma corrente, através de perfis de corrente. O diálogo é exibido na Figura 52.

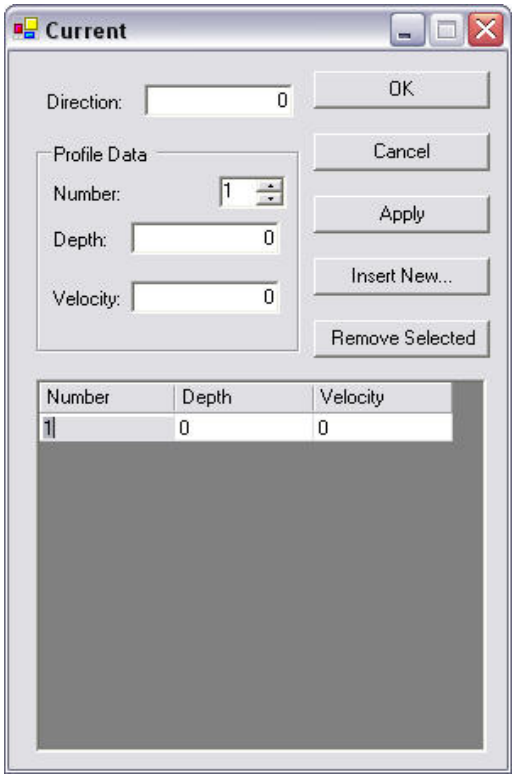


Figura 52 – Diálogo de inserção de corrente

Nesse diálogo insere-se a direção da corrente e modifica-se o perfil da corrente da mesma forma que se modifica uma propriedade ou um *keypoint*. Insere-se para cada perfil a profundidade e a velocidade nessa propriedade. Nesse diálogo estão presentes os botões *Insert New..., Remove, Cancel, OK* e *Apply* como nos diálogos anteriormente descritos.

III.4.3. Menu Simulation Parameters

O menu *Simulation Parameters* é mostrado na Figura 53. Esse menu permite o acesso às configurações de análise.

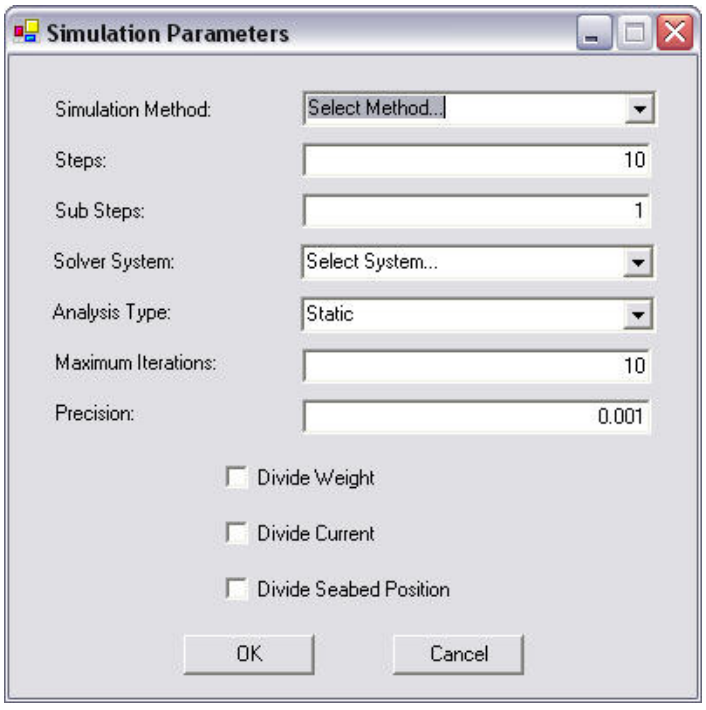


Figura 53 – Diálogo de parâmetros de simulação

Nesse diálogo pode-se selecionar o tipo de método a ser utilizado: *tangent stiffness*, *restauration method* e *incremental restauration method*, bem como o número de *steps* e *substeps* a serem utilizados. Pode-se escolher o tipo de *solver* do sistema entre Cholesky e LDLT.

Dentre as demais opções, pode-se escolher o tipo de análise que permite apenas a análise estática, número máximo de iterações e precisão. Pode-se ainda serem ajustadas algumas *flags* que podem facilitar a resolução do caso em estudo.

III.4.4. Menu XML Data

O menu *XML Data* é mostrado na Figura 54. Esse menu permite que dados sejam importados e exportados no formato XML.

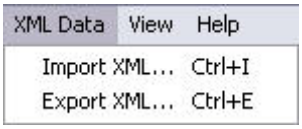


Figura 54 – Menu XML Data

- ***Comando Import XML:***

O comando Im*port XML* permite que o usuário importe os dados para de arquivo XML, abrindo um diálogo semelhando ao da Figura 33. Nesse caso, todos os dados são lidos e

armazenados no arquivo aberto atualmente e podem ser modificados, facilitando assim a geração de diversos casos com pequenas modificações.

- ***Comando Export XML:***

O comando Ex*port XML* permite que o usuário exporte os dados para um arquivo XML, abrindo um diálogo semelhando ao da Figura 34. Se esse arquivo exportado contém todos os dados necessários para a análise, pode alimentar o módulo de análise estática e posteriormente, com a saída de outro arquivo *XML* com os resultados da simulação, é possível alimentar o pós-processador.

III.4.5. Menu View

O menu *View* é mostrado na Figura 55 e na Figura 56. Esse permite executar as mesmas tarefas que os botões da barra de visualização, com exceção da mudança de cabos e alternar entre malha e geometria.

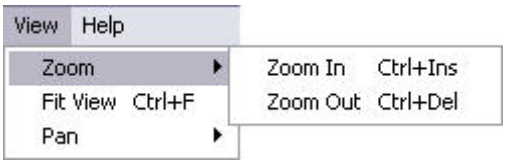


Figura 55 – Menu View com Zoom expandido

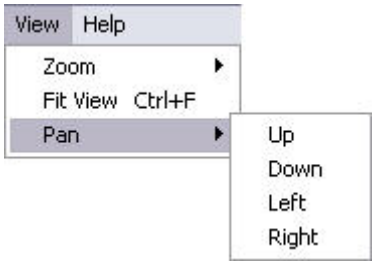


Figura 56 – Menu View com Pan expandido

- ***Comandos Zoom – Zoom In e Zoom – Zoom Out:***

O comando *Zoom* permite que o usuário se aproxime ou se afaste do desenho do cabo selecionado, facilitando a verificação da malha e da geometria. A Figura 57 mostra um exemplo quando clica-se em *Zoom In*, enquanto a Figura 58 mostra o mesmo exemplo quando em *Zoom Out*.

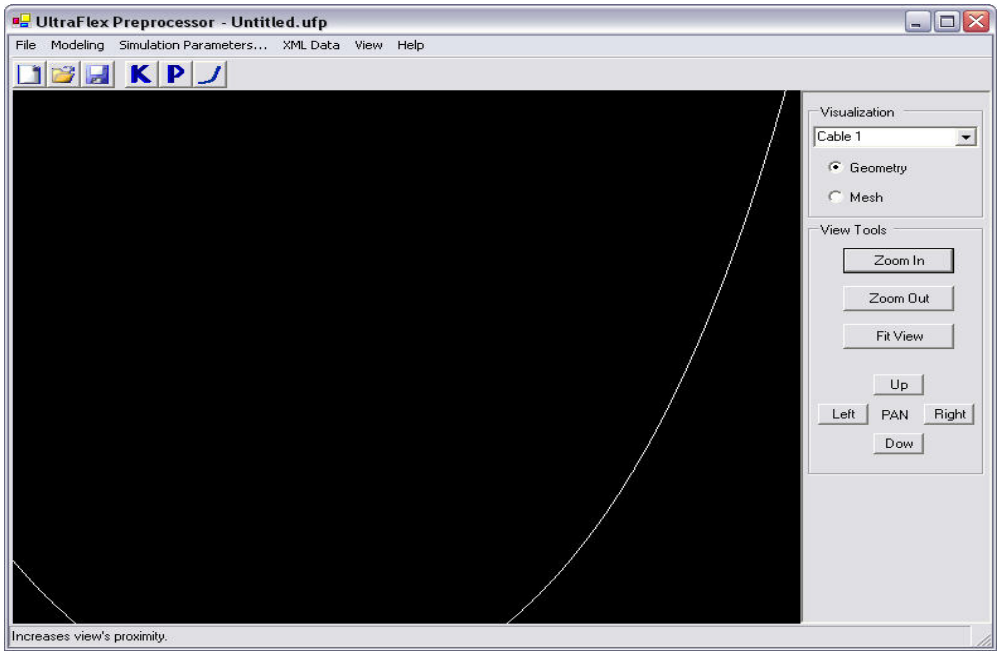


Figura 57 – Zoom In de cabo

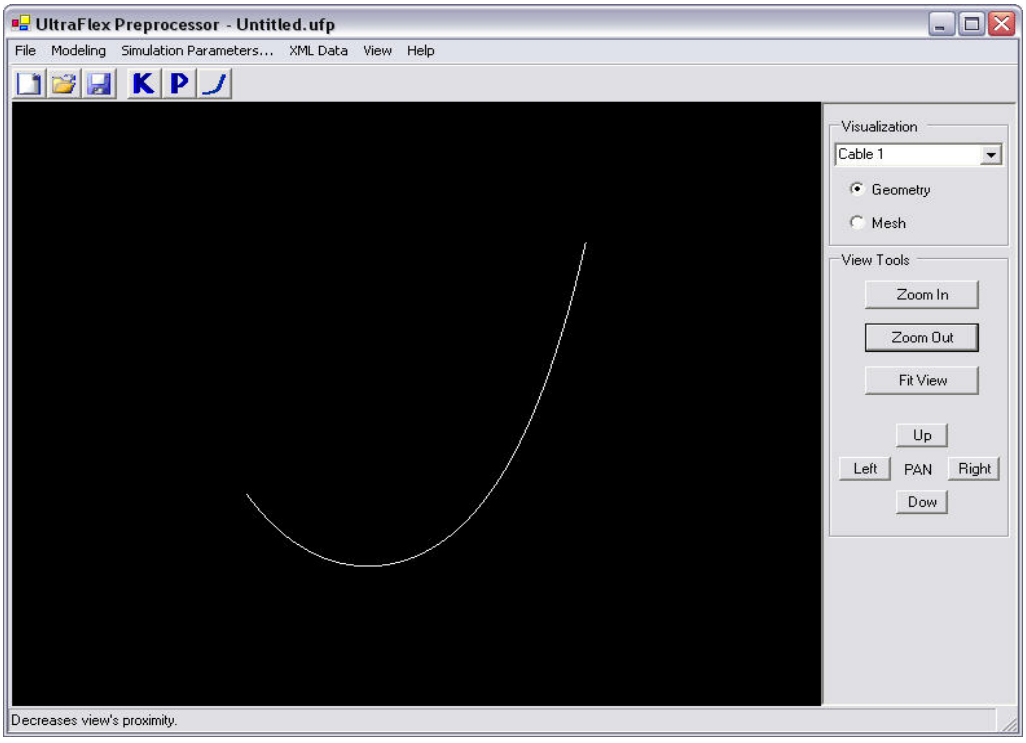


Figura 58 – Zoom Out de cabo

- ***Comando Fit View:***

O comando *Fit View* faz com que o desenho do cabo seja ajustado à área de visualização atual, mantendo as relações de proporção do desenho. A Figura 59 mostra o resultado do comando *Fit View*.

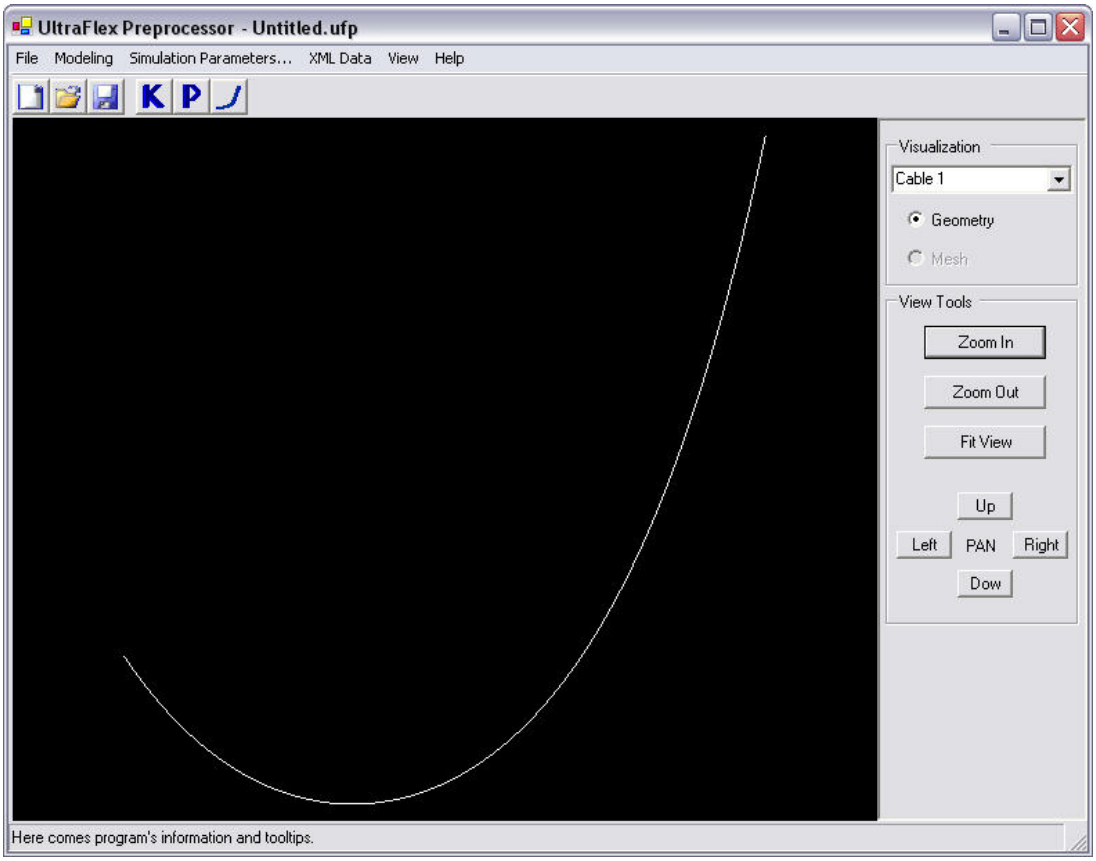


Figura 59 – Efeito do Fit View de um cabo

- ***Comandos Pan – Left, Pan – Right, Pan – Up e Pan – Down:***

Os comandos do grupo *Pan* permitem que o usuário visualize o cabo movimentando a janela de visualização. Clicando-se em cada uma das opções, a janela de visualização é movimentada. A Figura 60 mostra o efeito do *Pan Left*, a Figura 61 mosta o efeito do *Pan Right*, a Figura 62 o efeito do *Pan Up*, e finalmente a Figura 63 o efeito do *Pan Down*.

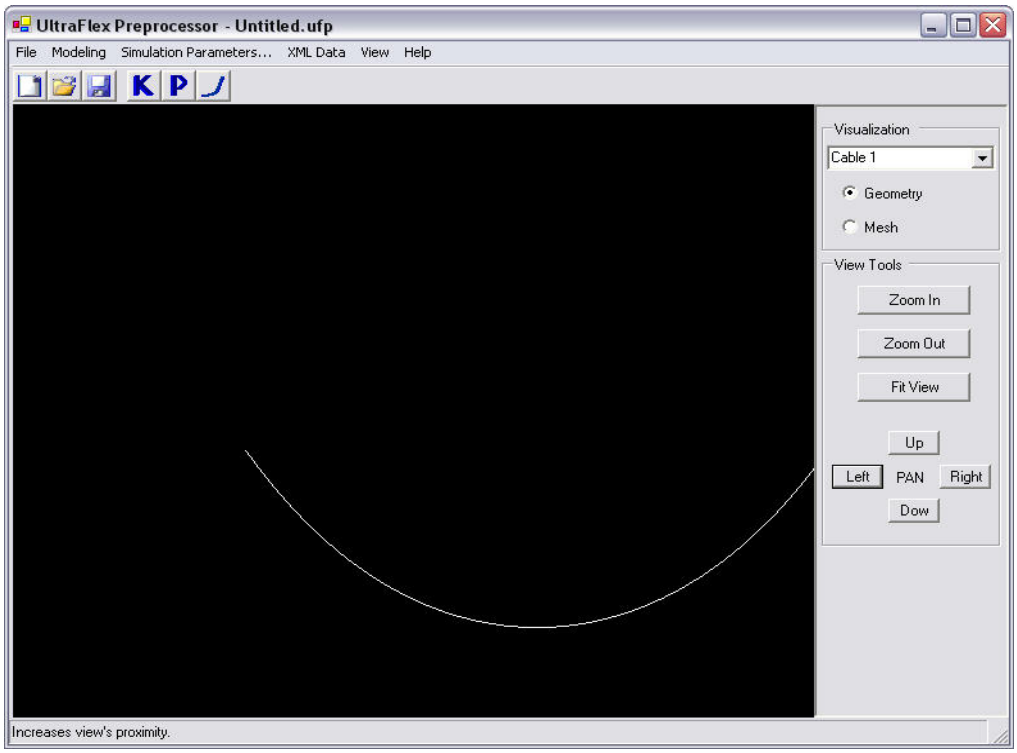


Figura 60 – Efeito do Pan Left

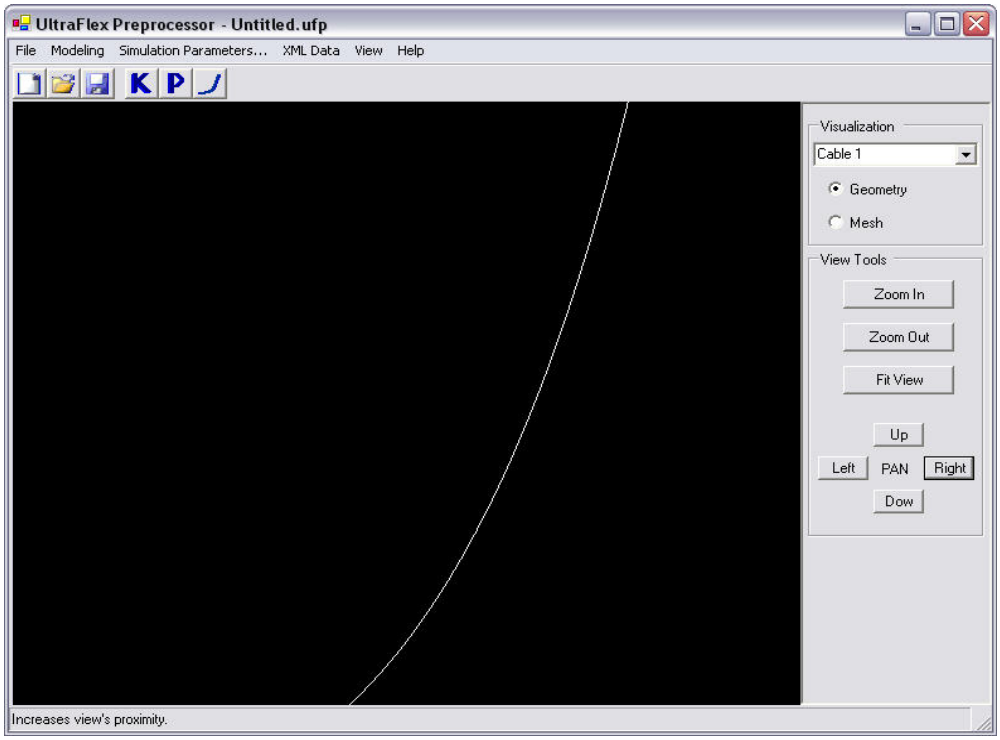


Figura 61 – Efeito do Pan Right

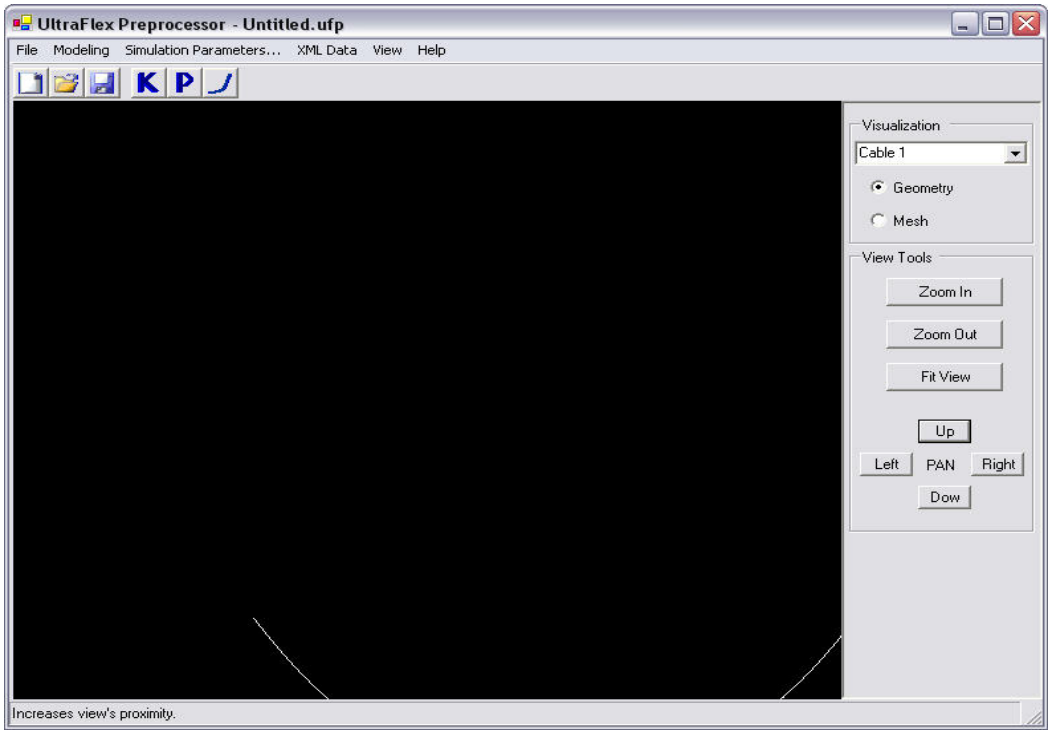


Figura 62 – Efeito do Pan Up

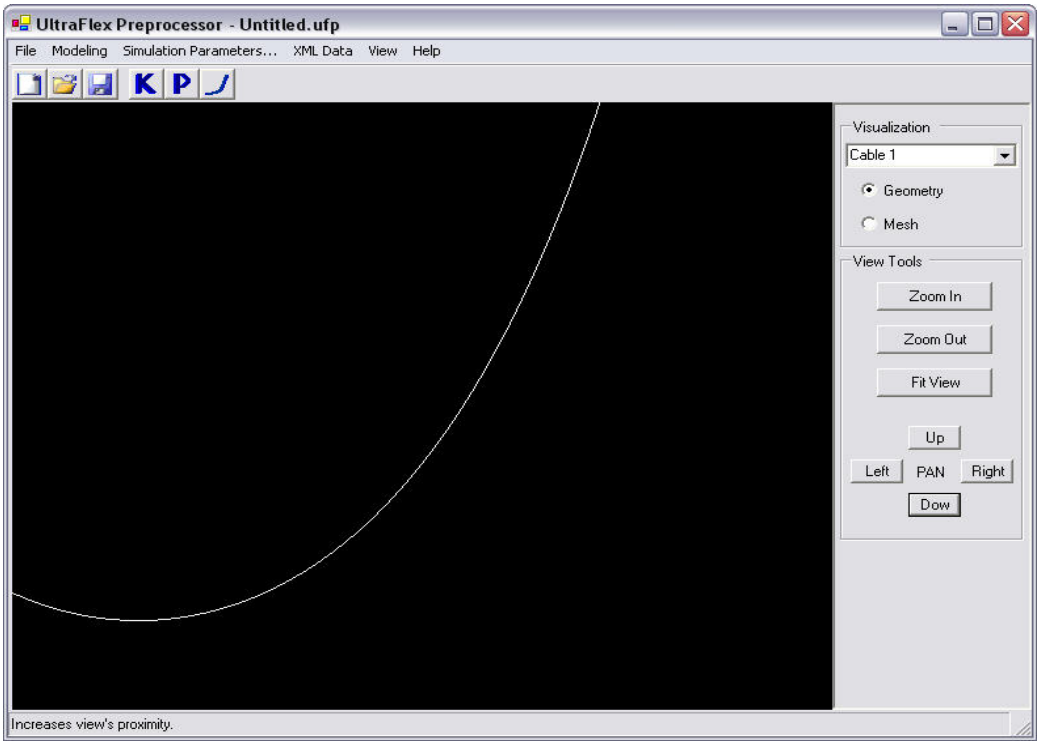


Figura 63 – Efeito do Pan Down

III.4.6. Menu Help

O menu *Help* é mostrado na Figura 64.Permite acesso aos dados do programa, através do diálogo *About*.

Figura 64 – Menu Help

- ***Comando About:***

O comando *About* exibe a janela sobre do programa, como pode ser visto na Figura 65.

Figura 65 – Diálogo Sobre

CAPÍTULO IV – PÓS-PROCESSADOR

IV.1 Especificações Técnicas

Da mesma maneira que os módulos anteriores, o pós-processador possui requisitos específicos. Nesse módulo, as especificações a serem atendidas são:

- A criação de um ambiente gráfico, não necessariamente multiplataforma, com o qual seja possível a leitura de um arquivo de entrada do núcleo do ambiente, sendo possível a visualização de gráficos de interesse do usuário.
- O software necessita ser "*user-friendly*", facilitando a visualização e geração dos gráficos desejados.
- Deve possuir funções de leitura e gravação de dados em XML, para que não seja necessária nenhuma intervenção do usuário no arquivo.
- Deve-se optar por uma linguagem de programação e bibliotecas gráficas que apresentem um compromisso entre velocidade e requisitos de *hardware* compatíveis e que possam ser utilizadas na maioria dos computadores pessoais atuais.

IV.1.1 Linguagem de Programação

Nesse módulo também se optou pela linguagem de programação C#, já que ela mostra as vantagens descritas no item III.1.1, referente à escolha da linguagem para o pré-processador. Um outro ponto que levou a escolha dessa linguagem foi a possibilidade de reutilizar tanto o

conhecimento adquirido com o desenvolvimento do pré-processador, bem como com o possível reaproveitamento de código e funções de leitura e gravação de XML.

Um outro aspecto a ser levantado é a utilização de bibliotecas, já que estas diferem das utilizadas no pré-processador. Escolheu-se a biblioteca Nevron® para a geração dos gráficos por ser de fácil utilização e por estar disponível para uso em projetos do laboratório NDF, do Departamento de Engenharia Mecânica da Escola Politécnica da Universidade de São Paulo.

IV.2 Descrição

O pós-processador é composto de uma janela principal na qual o usuário pode escolher o tipo de visualização de dados desejada, tanto na forma de uma tabela quanto na forma de um gráfico, escolhendo as colunas (ou os eixos, no caso do gráfico) desejadas para essa visualização. O usuário ainda tem a possibilidade de criar uma cópia do gráfico em uma janela separada, com o intuito de comparar diversos gráficos do mesmo caso.

Os dados são importados de arquivos XML e somente após a abertura de um desses arquivos é que a janela principal, exibida na Figura 66 torna-se operacional.

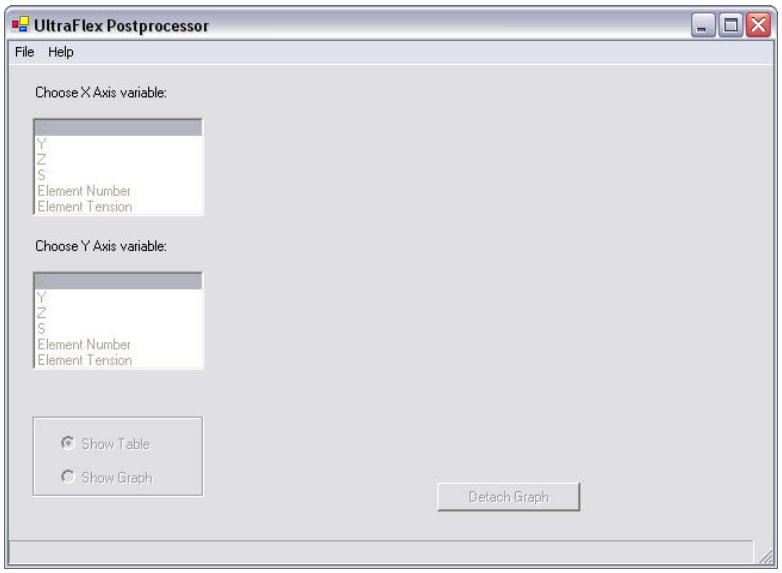
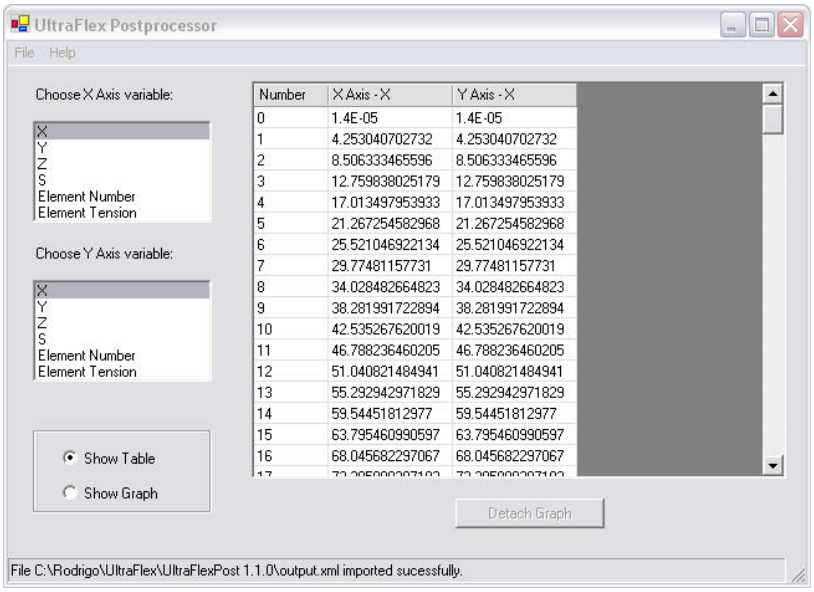


Figura 66 – Janela Principal do Pós-processador sem nenhum arquivo de entrada carregado

A entrada do arquivo a ser visualizado é feita através do menu File, escolhendo-se a opção "Import XML Data...". É possível reiniciar-se o processo selecionando-se a opção "New..." no mesmo menu. Uma vez que os dados foram devidamente carregados, é possível visualizar-se uma tabela com os dados dos eixos escolhidos (cujo padrão é X por X), como exibidos na Figura 67.



Number	X Axis - X	Y Axis - X
0	1.4E-05	1.4E-05
1	4.253040702732	4.253040702732
2	8.506333465596	8.506333465596
3	12.759838025179	12.759838025179
4	17.013497953933	17.013497953933
5	21.267254582968	21.267254582968
6	25.521046922134	25.521046922134
7	29.77481157731	29.77481157731
8	34.028482664823	34.028482664823
9	38.281991722894	38.281991722894
10	42.535267620019	42.535267620019
11	46.788236460205	46.788236460205
12	51.040821484941	51.040821484941
13	55.292942971829	55.292942971829
14	59.54451812977	59.54451812977
15	63.795460990597	63.795460990597
16	68.045682297067	68.045682297067

Figura 67 – Pós com arquivo de entrada carregado

Com o arquivo de entrada carregado, é possível modificar-se a visualização dos resultados para gráficos em duas dimensões, clicando-se em “Show Graph” no canto esquerdo inferior. A Figura 68 mostra o mesmo caso anteriormente carregado quando selecionada essa opção.

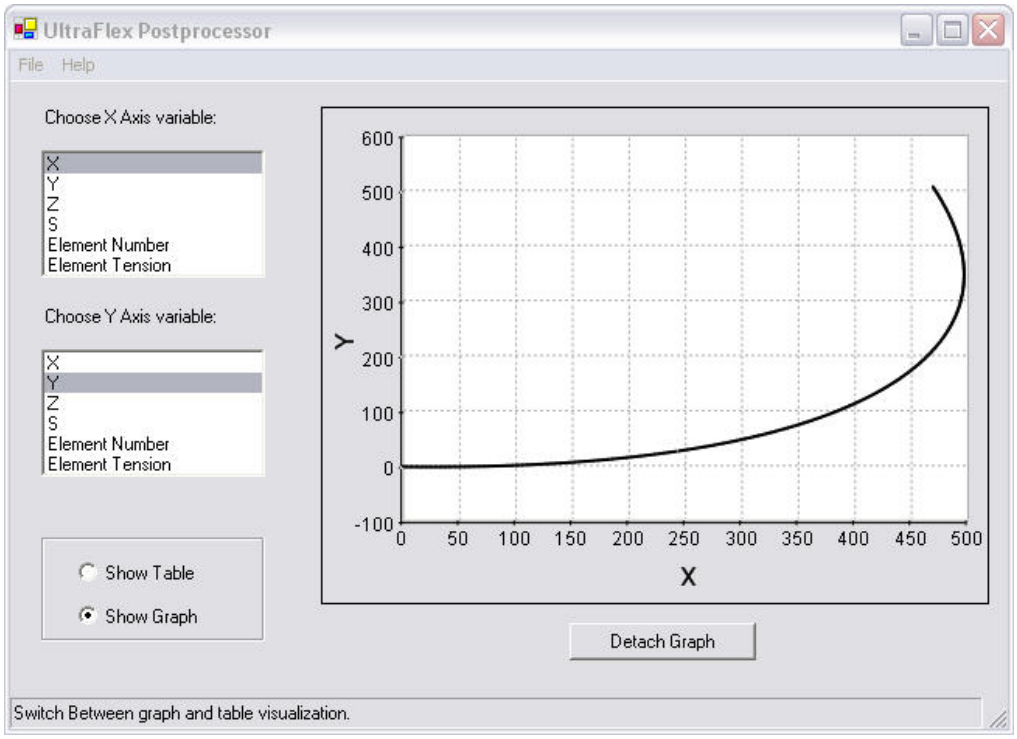


Figura 68 – Visualização de gráficos

As opções de gráficos possíveis são as mais comuns e necessárias para a validação e verificação dos modelos. Ainda ressalta-se a possibilidade da visualização de diversos gráficos simultaneamente, uma vez que os gráficos podem ficar em novas janelas. Clicando-se no botão “*Detach Graph*”, uma nova janela, similar à da Figura 69, será exibida. É possível alterar o gráfico do programa principal, ou mesmo alterar para a visualização dos dados na forma de tabela sem perder o conteúdo do gráfico desta nova janela.

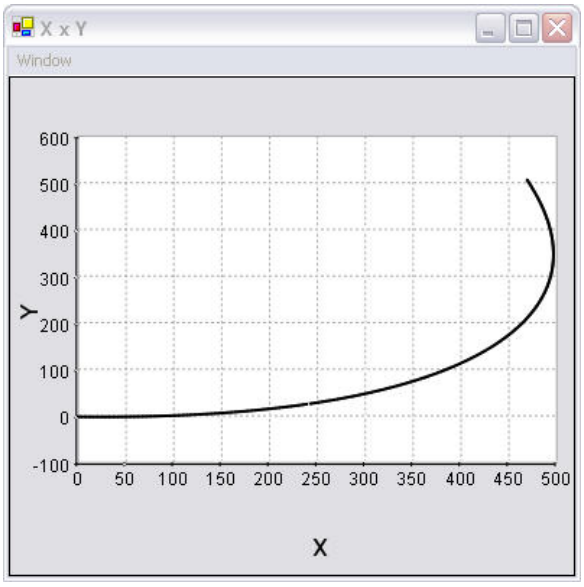


Figura 69 – Gráfico exibido em nova janela

Nesta janela ainda é possível ter acesso aos recursos de impressão e visualização de impressão, como pode ser visto na Figura 70, na Figura 71 e na Figura 72.

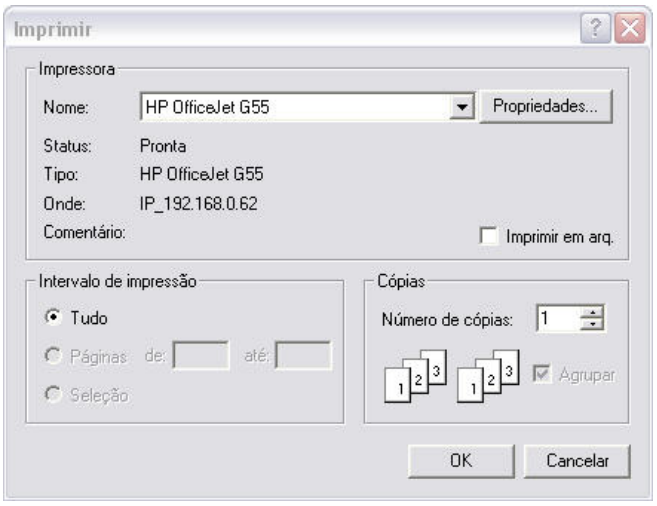


Figura 70 – Janela de Impressão

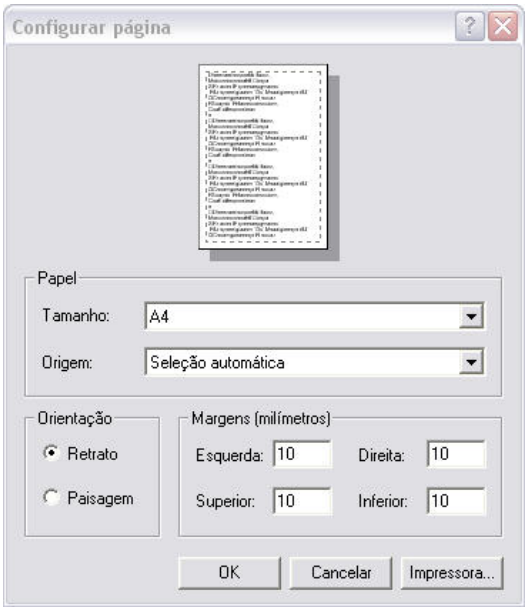


Figura 71 – Janela de configurações de impressão

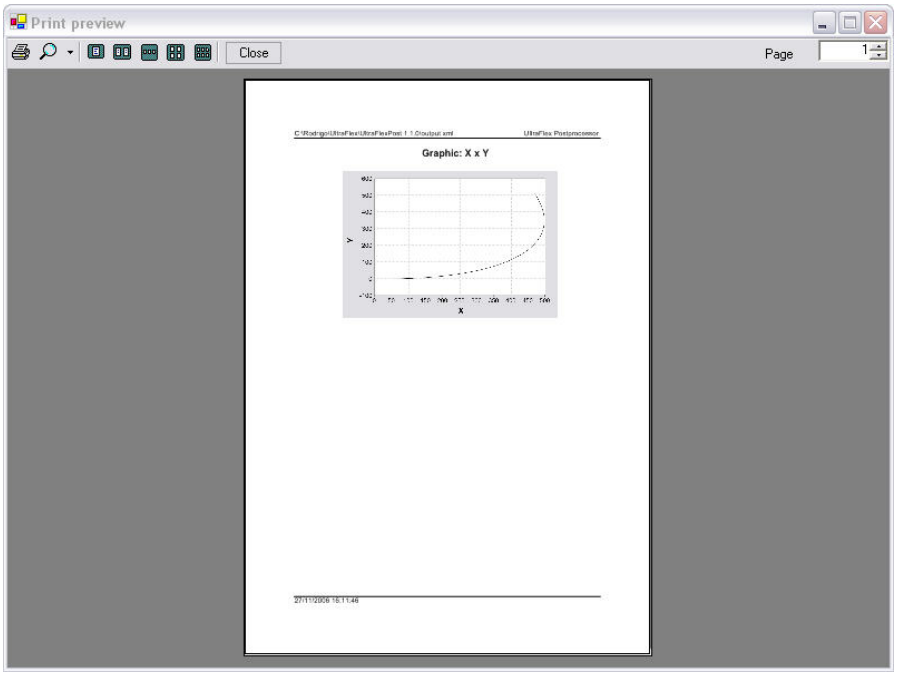


Figura 72 – Janela de visualização de impressão

Na Figura 73 tem-se uma possível configuração do software, exibindo diversos gráficos ao mesmo tempo e, assim, aproveitando ao máximo os recursos possíveis.

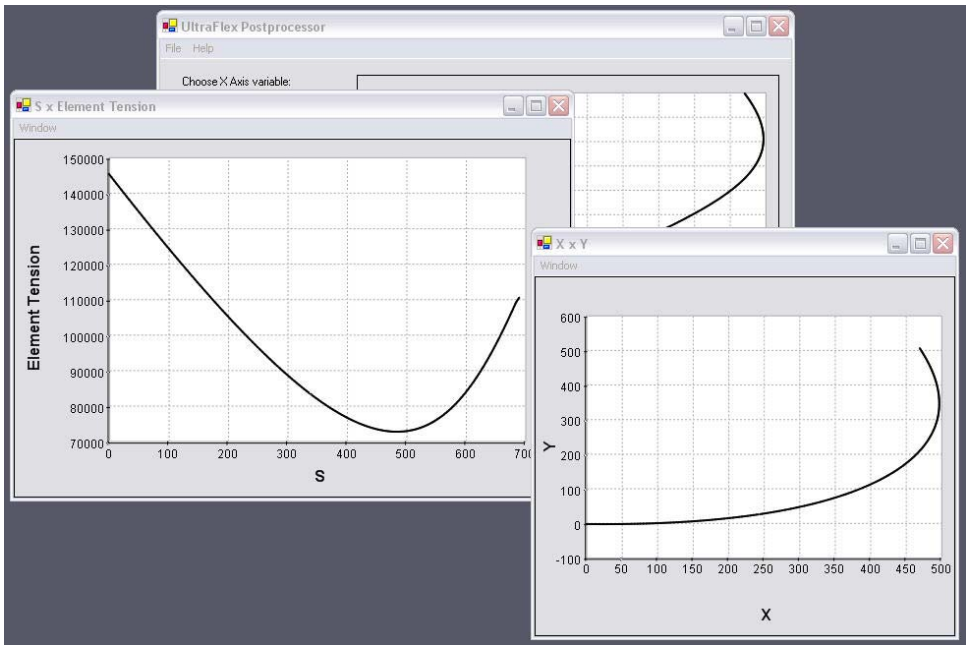


Figura 73 – Pós processador com diversas janelas de gráficos abertas

REFERÊNCIAS

[1] PATEL, M.H.; SEYED, F.B.; *Review of Flexible Riser Modeling and Analysis technique* **Engineering Structures**, Great Britain, v. 17, No 4, p. 293-304, 1995.

[2] MARTINS, C.A.; **Cabos Flexíveis Inextensíveis**. Apostila.

[3] BATHE, K. J.; **Finite Element Procedures**. New Jersey: Prentice-Hall Inc, 1996. 1037 p.

[4] PRESS,W.H.; TEUKOLSKY,S.A.;VETTERLING,W.T.; FLANNERY, B.P.; **Numerical Recipes in C**. 2nd Ed. New York: Cambridge University Press, 1992.994 p.

[5] GOLUB, G.H.; LOAN, C.F.V.; **Matrix Computations.** Third Edition. Baltimore: Johns Hopkins University Press, 1996. 694 p.

[6] PAULETTI, R.; **Análise Não Linear de Estruturas I**. Notas de Aula.

[7] PRZEMIENIECKI, J,S,; **Theory of Matrix Structural Analysis.** New York: Dover Publications Inc, 1985. 468 p.

[8] GERE, J.M.; **Mecânica dos Materiais**. São Paulo: Pioneira Thomson Learning, 2003. 689 p.

[9] FRANÇA L.N.F.; MATSUMARA A.Z.; **Mecânica Geral.** São Paulo: Editora Edgard Blücher LTDA, 2001. 235p.

[10] COOK, R.D.; **Concepts and applications of finite element analysis**. New York: Wiley, 1989. 630p.

[11] SCHILDT, H.; C++: **The Complete Reference**. Third Edition. McGraw-Hill, 1998. 1008 p.

[12] FÍSICA COMPUTACIONAL - *Professor Carlos Bertulani.* Disponível em <http://www.if.ufrj.br/teaching/compute/index.html>. Acesso em 12/06/2006.

[13] BARONE Jr, M.; **Álgebra Linear**. 3ª edição. São Paulo, 2001. 312 p.

[14] TEMPLEMAN, J. **Microsoft Visual C++ .NET step by step.** 1ª edição, Microsoft Press, 581 p., 2003.

[15] SHARP, J. **Microsoft Visual C# .NET step by step**. 1ª edição, Microsoft Press, 635 p., 2003.

[16] SCHILDT, H. **The Complete Reference: C++**. Third Edition, McGraw-Hill, 1998. 995p.

[17] **OpenGL® – The Industry for High Performance Graphics.** Site oficial do *OpenGL®*. Disponível em <http://www.opengl.org>. Acessado em 15 de março de 2006.

[18] **NeHe Productions.** Disponível em <http://nehe.gamedev.net>. Acessado em 23 de março de 2006.

[19] **OpenGL® Reference Manual. OpenGL® Architecture Review Board.** Addison−Wesley Publishing Company. Silicon Graphics, 1994.

[20] COHEN, M.; MANSOUR, I. H. **OpenGL® – Uma abordagem prática e objetiva.** Novatec, 2006. 478p.

[21] **The Code Project – Free Source Code and Tutorials**. Disponível em <http://www.codeproject.com/>. Acessado em 12 de março de 2006.

[22] **TAO Framework**. Disponível em <http://www.taoframework.com/> . Acessado em 20 de abril de 2006.

[23] RUGGIERO, M.; LOPES, V**. Cálculo Numérico: Aspectos teóricos e Computacionais.** 2ª Edição, Makron Books, p. 192-200, 1997.

APÊNDICE A – ROTAÇÃO DE SISTEMAS DE COORDENADAS [13]

As formulações dos elementos aqui desenvolvidas são feitas baseadas no sistema de coordenadas local dos mesmos. Sendo assim, a matriz de rigidez montada não é adequada para qualquer sistema de coordenadas, sendo, portanto necessário realizar uma rotação sobre ela. Executando tal procedimento é possível escrever a matriz de rigidez de cada elemento com base no sistema de coordenadas global.

Sendo V um vetor escrito no sistema de coordenadas global, e $\overline{\mathbf{V}}$ o mesmo vetor escrito no sistema de coordenadas local, existe uma matriz T tal que:

$$\mathbf{V} = \mathbf{T}\overline{\mathbf{V}} \tag{A.1}$$

Onde T é a matriz identidade alterada pelos valores dos cossenos diretores dos componentes de $\overline{\mathbf{V}}$ medidos no sistema global de coordenadas.

Foram desenvolvidas as matrizes de rotação para sistemas de coordenadas bi e tridimensionais. A equação (A.2) encontra a matriz bidimensional para a rotação entre os sistemas de coordenadas da Figura 74.

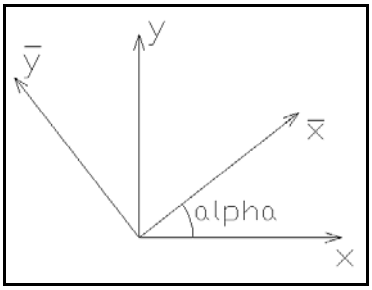


Figura 74 – Rotação de um sistema de coordenadas bidimensional

$$\mathbf{T} = \begin{bmatrix} \cos(\alpha) & -sen(\alpha) \\ sen(\alpha) & \cos(\alpha) \end{bmatrix} \tag{A.2}$$

Para o caso tridimensional foi utilizado um método algébrico, para se determinar os valores dos cossenos diretores que aparecem na matriz de rotação.

Sendo $(\vec{i},\vec{j},\vec{k})$ uma base ortogonal, mas não necessariamente ortonormal de um sistema de coordenadas e denotando os valores das coordenadas dos dois nós que formam um elemento (x_1,y_1,z_1) e (x_2,y_2,z_2), pode-se definir:

$$\vec{i} = (x_2 - x_1, y_2 - y_1, z_2 - z_1)$$

O valor do produto escalar $\vec{i} \bullet \vec{j} = 0$ pois a base é ortogonal. Sendo $\vec{j} = (a,b,c)$ onde *a*,*b* e *c* são constantes reais a serem determinadas, desenvolve-se o produto escalar:

$$\vec{i} \bullet \vec{j} = \left[(x_2 - x_1)a + (y_2 - y_1)b + (z_2 - z_1)c\right] = 0 \qquad \text{(A.3)}$$

Admitindo arbitrariamente os valores de $a = 0$ e $b = 1$, pois esses definirão as posições dos eixos ortogonais à direção do elemento, que são absolutamente arbitrários. Agora é possível determinar o valor de *c* com a equação (A.3). Encontra-se assim o valor:

$$c = \frac{(y_1 - y_2)}{(z_2 - z_1)} \qquad \text{(A.4)}$$

Basta agora determinar uma expressão para o vetor $\vec{k}$. Fazendo o produto vetorial dos outros dois vetores é possível obter a equação (A.5).

$$\vec{k} = \vec{i} \times \vec{j} = \left(-\frac{(y_2 - y_1)^2}{(z_2 - z_1)} - (z_2 - z_1), \frac{(y_2 - y_1)(x_2 - x_1)}{(z_2 - z_1)}, (x_2 - x_1) \right) \qquad \text{(A.5)}$$

partir dos vetores escritos é possível determinar os cossenos diretores entre as coordenadas globais e locais e montar a matriz de rotação tridimensional (equação (A.6)):

$$\cos(x,\overline{x}) = \frac{\vec{i} \bullet (1,0,0)}{\left\|\vec{i}\right\|},\ \cos(y,\overline{x}) = \frac{\vec{i} \bullet (0,1,0)}{\left\|\vec{i}\right\|},\ \cos(z,\overline{x}) = \frac{\vec{i} \bullet (0,0,1)}{\left\|\vec{i}\right\|}$$

$$\cos(x,\bar{y}) = \frac{\vec{j} \bullet (1,0,0)}{\left\|\vec{j}\right\|},\ \cos(y,\bar{y}) = \frac{\vec{j} \bullet (0,1,0)}{\left\|\vec{j}\right\|},\ \cos(z,\bar{y}) = \frac{\vec{j} \bullet (0,0,1)}{\left\|\vec{j}\right\|}$$

$$\cos(x,\bar{z}) = \frac{\vec{k} \bullet (1,0,0)}{\left\|\vec{k}\right\|},\ \cos(y,\bar{z}) = \frac{\vec{k} \bullet (0,1,0)}{\left\|\vec{k}\right\|},\ \cos(z,\bar{z}) = \frac{\vec{k} \bullet (0,0,1)}{\left\|\vec{k}\right\|}$$

$$\mathbf{T} = \begin{bmatrix} \cos(x,\bar{x}) & \cos(x,\bar{y}) & \cos(x,\bar{z}) \\ \cos(y,\bar{x}) & \cos(y,\bar{y}) & \cos(y,\bar{z}) \\ \cos(z,\bar{x}) & \cos(z,\bar{y}) & \cos(z,\bar{z}) \end{bmatrix} \tag{A.6}$$

APÊNDICE B – INTEGRAÇÃO NUMÉRICA: MÉTODO DOS TRAPÉZIOS [13]

O método dos Trapézios é um método de integração baseado na aproximação da curva a ser integrada por segmentos de retas entre dois pontos consecutivos. A Figura 75 mostra como é calculada a integração pelo Método dos Trapézios.

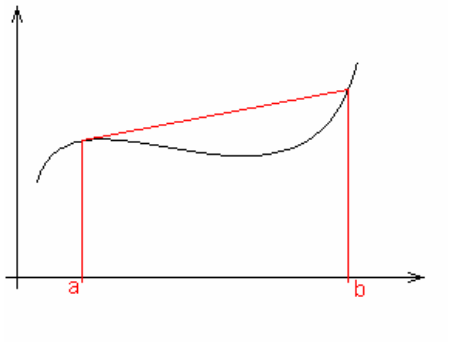


Figura 75 – Método dos Trapézios para intervalo de integração [a,b]

Aproxima-se a curva por uma reta e a área formada entre esta e o eixo, cuja figura geométrica dá nome ao método, é a integral aproximada do mesmo. Para um caso unidimensional em que se possui a função *f(x)* e cujos extremos de integração são *a* e *b*, a integral pelo Método dos Trapézios é dada por (B.1).

$$\int_a^b f(x) \approx \frac{b-a}{2}\left(f(a)+f(b)\right) \tag{B.1}$$

Para obter-se um valor mais realístico da integral, deve-se utilizar este método em partes, ou seja, aproximando pequenos trechos da função por integrais desse tipo, como na Figura 76.

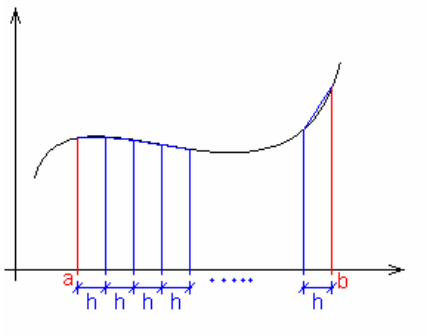


Figura 76 – Método do Trapézio com *n* divisões e passo *h*

Definindo o passo como a diferença entre os extremos de integração dividida pelo número de divisões desejadas *n* – equação (B.2) – esse problema pode ser resolvido.

$$h = \frac{b-a}{n} \tag{B.2}$$

Assim, tem-se então que a integral pelo Método dos Trapézios para *n* divisões iguais – equação (B.3).

$$\int_a^b f(x) = \frac{h}{2}\left(f(a) + f(b) + \sum_{i=1}^{n-1} f(a+ih) \right) \tag{B.3}$$

APÊNDICE C – INTEGRAÇÃO NUMÉRICA: MÉTODO DE ROMBERG

O Método de Romberg é um tipo de implementação de método de integração numérica que se utiliza de uma malha de integrações feitas através do Método dos Trapézios. Assim, o método calcula a integral do Método dos Trapézios com um número crescente de divisões e verifica a convergência, além de poder trabalhar com uma precisão de erro e um critério de parada de iterações. O método ainda fornece como resultado da integração numérica uma composição do valor da integral, como é exibido a seguir:

$$\int_a^b f(x) \approx I_{j,k} = \frac{4^{k-1} I_{j+1,k-1} - I_{j,k-1}}{4^{k-1} - 1}$$

Onde,

o índice *j* é o número de subdivisões no Método dos Trapézios.

o índice *k* indica o ordem estimada do erro da aproximação.

A implementação do método pode ser feita da seguinte maneira:

- Calcule a integral pelo Método dos Trapézios $I_{1,1}$.
- Comece com as subdivisões e calcule $I_{i,1}$, usando o Método dos Trapézios para 2^i intervalos.
- Calcule em subdivisões com *k* começando de *2* e indo até $i + 1$ a integral $I_{j,k}$ com $j = 2 + i - k$ e utilize a regra de Romberg descrita anteriormente.
- Finalize as subdivisões caso atingido o número máximo de iterações ou o erro.

APÊNDICE D – EXPRESSÃO PARA O CÁLCULO DA CURVATURA GEOMETRICAMENTE EXATA

Tomando um elemento infinitesimal de pórtico, ilustrado na Figura 77 é possível escrever a relação geométrica da tangente do ângulo de inclinação, ilustrado na mesma figura.

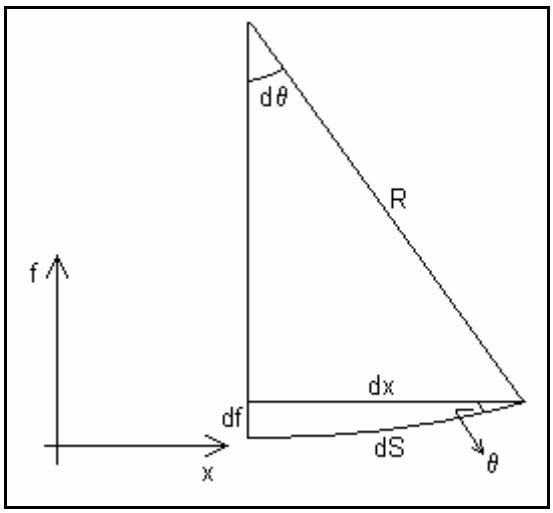


Figura 77 – Elemento diferencial de Pórtico

$$\tan(\theta) = \frac{df}{dx} \tag{D.1}$$

Diferenciando-se ambos os lados da equação (D.1), obtém-se:

$$\sec^2(\theta) d\theta = \frac{d^2 f}{dx^2} dx \tag{D.2}$$

Substituindo as equações (D.1) e (D.2) na relação trigonométrica: $\sec^2(\theta) = 1 + \tan^2(\theta)$, obtém-se:

$$\frac{d^2 f}{dx^2} \frac{dx}{d\theta} = 1 + \left(\frac{df}{dx} \right)^2 \tag{D.3}$$

É possível escrever outra relação geométrica, a partir do Teorema de Pitágoras, aplicado na geometria do elemento de pórtico diferencial:

$$df^2 + dx^2 = dS^2 \tag{D.4}$$

Mas também:

$$\begin{cases} \sec(\theta) = \dfrac{1}{\cos(\theta)} = \dfrac{dS}{dx} \\ \sec^2(\theta) = \left(\dfrac{dS}{dx}\right)^2 = 1 + \left(\dfrac{df}{dx}\right)^2 \Leftrightarrow dS^2 = dx^2\left(1 + \left(\dfrac{df}{dx}\right)^2\right) \end{cases} \tag{D.5}$$

Pode-se relacionar *dS* com *dx* da seguinte forma:

$$dS = dx\left(1 + \left(\frac{df}{dx}\right)^2\right)^{1/2} \tag{D.6}$$

Como $dS = Rd\theta$, pode-se escrever, isolando-se $d\theta$ na equação (D.3):

$$\frac{dS}{R} = \frac{\dfrac{d^2 f}{dx^2}dx}{\left(1 + \left(\dfrac{df}{dx}\right)^2\right)} \tag{D.7}$$

Substituindo (D.6) em (D.7), obtém-se:

$$\frac{dx\left(1 + \left(\dfrac{df}{dx}\right)^2\right)^{1/2}}{R} = \frac{\dfrac{d^2 f}{dx^2}dx}{\left(1 + \left(\dfrac{df}{dx}\right)^2\right)} \Leftrightarrow \left(\frac{1}{R}\right) = \kappa = \frac{\dfrac{d^2 f}{dx^2}}{\left(1 + \left(\dfrac{df}{dx}\right)^2\right)^{3/2}} \tag{D.8}$$

Onde κ é a curvatura do pórtico.

APÊNDICE E – MÉTODO DE NEWTON APLICADO À RESOLUÇÃO DE SISTEMAS NÃO-LINEARES

O método de Newton baseia-se no algoritmo desenvolvido em Rugierro, M. e Lopes, V., 1997 ([23]) As autoras descrevem que um sistema de equações, pode ser escrito na forma exibida em (E.1).

$$\mathbf{F(x) = 0} \tag{E.1}$$

Para resolver o sistema não linear, inicialmente é feita uma expansão em série de Taylor de cada uma das funções ao redor das estimativas de suas raízes ($x^{(k)}$), uma vez que as mesmas são desconhecidas. Para uma função qualquer f_i, calculada em um ponto x ao redor de $x^{(k)}$, tem-se a expansão como em (E.2).

$$f_i(x) = f_i(x^{(k)}) + \nabla f_i(c_i)^T (x - x^{(k)}) \quad i = 1,....,n \tag{E.2}$$

Onde c_i é um valor desconhecido entre x e $x^{(k)}$. Ao invés de calcular $\nabla f_i(c_i)$ mas calculando $\nabla f_i(x^{(k)})$, obtém-se uma aproximação linear para a função $f_i(x)$, como descrito na equação (E.3).

$$f_i(x) = f_i(x^{(k)}) + \nabla f_i(x^{(k)})^T (x - x^{(k)}) \quad i = 1,....,n \tag{E.3}$$

Aplicando o método para todas as funções $f_i(x)$, obtém-se **F(x)** como na equação (E.4).

$$\mathbf{F(x) \approx F(x^{(k)}) + J(x^{(k)})(x - x^{(k)})} \tag{E.4}$$

Onde **J** é a matriz Jacobiana. Agora impõe-se que **F(x)** seja identicamente igual ao vetor nulo e com isso obtém-se uma aproximação para o vetor **x**.

$$\mathbf{0 = F(x^{(k)}) + J(x^{(k)})(x - x^{(k)})} \tag{E.5}$$

O algoritmo, portanto, consiste em, dado uma estimativa inicial para as raízes $\mathbf{x^{(k)}}$:

- Calcular o vetor $\mathbf{F(x^{(k)})}$ e $\mathbf{J(x^{(k)})}$.
- Obter a solução do sistema linear $\mathbf{J(x^{(k)})\Delta}x = \mathbf{F(x^{(k)})}$ em $\mathbf{\Delta x}$.
- Somar os valores de $\mathbf{\Delta x}$ à estimativa inicial e recomeça-se o processo tendo como estimativa o novo $\mathbf{x}$.

O processo iterativo termina quando o erro residual entre a diferença entre dois vetores **F** consecutivos for menor que a precisão indicada quando da convergência ou quando atinge um limite máximo de iterações, que indica que o número de iterações não permitiu obter-se a resposta dentro da precisão estabelecida.