

# Ambiente de programação científica orientada a objetos NeoPZ: Uma introdução para novos alunos

Francisco T. Orlandini  
LabMeC  
UNICAMP

Supervisionado por Nathan Shauer e Philippe Devloo  
francisco.orlandini@gmail.com

Outubro 2014

## **Resumo**

Neste tutorial buscamos apresentar de forma simples os conceitos fundamentais para um estudante ingressante no LabMeC, assim como algumas das ferramentas que virão a ser utilizadas no futuro. No decorrer de três meses é esperado que o aluno ambiente-se com a linguagem de programação C++, com o software Mathematica e com as ideias centrais em torno das quais se baseia o Método dos Elementos Finitos, além de uma visualização e experimentação inicial com o pacote NeoPZ.

## Conteúdo

<b>1</b>	<b>Introdução</b>	<b>3</b>
1.1	Motivação . . . . .	3
1.2	NeoPZ . . . . .	3
1.3	Organização do tutorial e cronograma proposto . . . . .	4
<b>2</b>	<b>Programação orientada a objeto e introdução a C++</b>	<b>6</b>
2.1	Programação orientada a objeto . . . . .	6
2.2	A linguagem de programação C++ . . . . .	6
2.3	Exercício . . . . .	6
2.3.1	Ambientação . . . . .	6
2.3.2	Estudo prático da linguagem C++ . . . . .	7
<b>3</b>	<b>O Método dos Elementos Finitos (FEM): Uma introdução</b>	<b>9</b>
3.1	Oden - Capítulo I . . . . .	9
3.2	Opcional: Oden - Capítulo II . . . . .	10
<b>4</b>	<b>Introdução ao software Mathematica: Criando um primeiro código utilizando o FEM</b>	<b>11</b>
4.1	Implementação das funções chapéu . . . . .	11
4.2	Implementação das Estruturas de Dados . . . . .	12
4.3	Resolução do Problema . . . . .	12
4.4	Opcional: Implementação de Funções Lagrangianas . . . . .	12
4.4.1	Elemento mestre . . . . .	13
4.4.2	Funções de Forma Lagrangianas . . . . .	13
4.4.3	Assemblagem . . . . .	13
4.4.4	Condições de Contorno . . . . .	14
<b>5</b>	<b>O pacote NeoPZ e a implementação do Problema Modelo</b>	<b>15</b>
5.1	Dependências . . . . .	15
5.2	Exemplo Simples . . . . .	15
<b>6</b>	<b>Bibliografia</b>	<b>16</b>
<b>A</b>	<b>Phillipe R. B. Devloo - Simulação Numérica</b>	<b>17</b>

# 1 Introdução

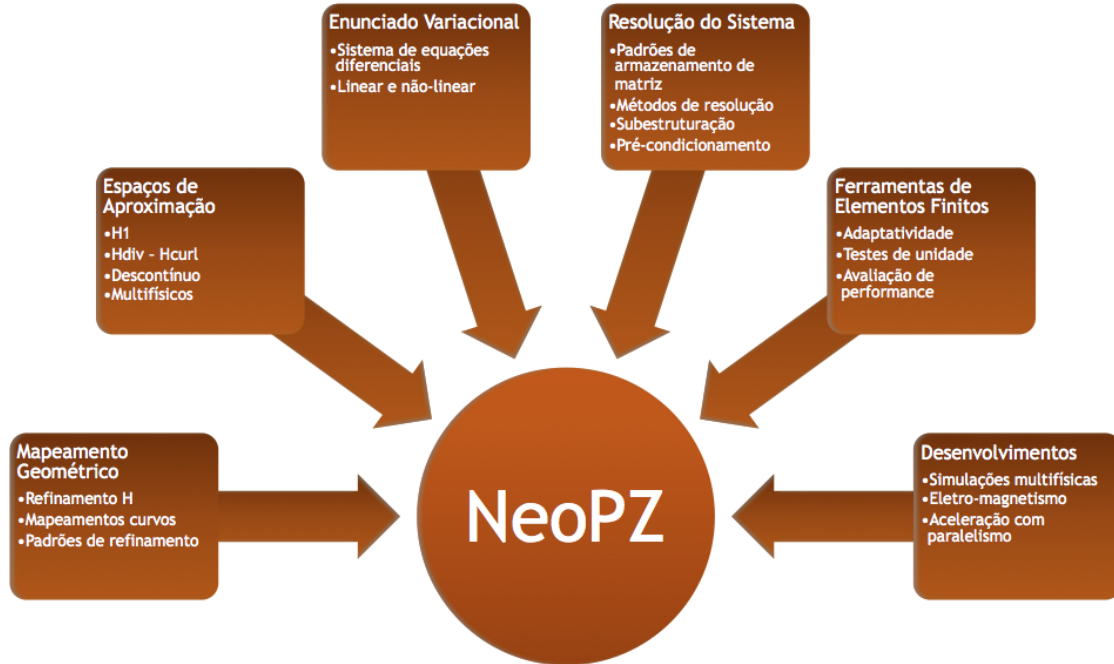
## 1.1 Motivação

Devido à grade dos cursos de graduação em Engenharia, é comum que o aluno ingressante no LabMeC não tenha conhecimento prévio na área em que virá a trabalhar. Este tutorial busca introduzir o aluno à ideia da implementação computacional do Método dos Elementos Finitos (FEM) distinguindo e esclarecendo os conceitos necessários para tal.

## 1.2 NeoPZ

Atualmente, existem diversos de pacotes que implementam o FEM disponíveis no mercado, dentre eles podem ser citados *Phoenix*, *Ansys*, *Sap2000*, *FlexPDE*, *Consol* e *Hermes*, dentre outros. O uso do pacote NeoPZ faz-se interessante pois proporciona uma maior liberdade ao usuário, por exemplo, quanto à liberdade quanto à representação de geometrias arbitrárias - a aproximação de um círculo por um conjunto de segmentos de retas pode, por exemplo, influenciar bastante na simulação de problemas eletromagnéticos. Também permite a resolução de qualquer equação diferencial descrevendo um fenômeno físico, podendo então ser aplicado nas mais diversas áreas do conhecimento. Os diferenciais mais importantes do ambiente NeoPZ em relação aos outros pacotes são:

- Diversos espaços de aproximação  $H^1$ ,  $L^2$ ,  $H^{div}$ ,  $H^{curl}$ , podendo também o usuário definir o seu próprio espaço de aproximação.
- Diversos métodos de resolução de Sistemas Lineares - Decomposição  $LU$ ,  $LDL^t$ , Cholesky. Métodos iterativos como Gradiente conjugado, *GMRES*, com ou sem pré-condicionador, dentre outros.
- Compila em qualquer plataforma - Windows, Linux, OSX
- HP adaptatividade em 1, 2 e 3D - Permite refinamento da malha somente onde este se faz necessário para obter uma melhor aproximação.



**Figura 1:** Visão geral da arquitetura NeoPZ

### 1.3 Organização do tutorial e cronograma proposto

As atividades propostas são divididas em quatro partes, e é esperado que o estudante leve três semanas para cada uma delas. Em caso de dificuldades, alguém do LabMeC deve ser avisado. A estrutura das partes é:

1. Introdução a C++
2. Introdução ao FEM
3. Introdução ao *Mathematica*
4. Introdução ao pacote NeoPZ

Na primeira parte é discutida a questão da programação orientada a objeto e a justificativa do emprego desta filosofia no pacote NeoPZ. Em seguida, são propostos dois pequenos exercícios para familiarizar o estudante com conceitos essenciais de C++.

Já na segunda parte, através da leitura de capítulos do livro *Finite Elements: An Introduction Volume I* (Oden et al., 1981), é esperado que o estudante adquira certa familiaridade com o FEM através da análise de problemas unidimensionais. Na terceira parte

o estudante deve realizar dois exercícios utilizando o software Mathematica e aplicar os conceitos vistos na leitura dos capítulos do livro. Finalmente, na quarta parte o aluno tem uma primeira experiência com pacote NeoPZ, observando a simulação da mesma equação diferencial analisada nos capítulos lidos de Oden et al. (1981) e podendo comparar os resultados com os obtidos no Mathematica.

## 2 Programação orientada a objeto e introdução a C++

### 2.1 Programação orientada a objeto

Conforme explicado em Deitel and Deitel (2012): “*Objects, or more precisely[...], the classes objects come from, are essentially reusable software components. There are date objects, time objects, audio objects, video objects, automobile objects, people objects, etc. Almost any noun can be reasonably represented as a software object in terms of attributes (e.g., name, color and size) and behaviours (e.g., calculating, moving and communicating). Software developers are discovering that using a modular, object-oriented design and implementation approach can make software-development groups much more productive than was possible with earlier popular techniques like “structured programming”—object-oriented programs are often easier to understand, correct and modify.*”. Para uma maior compreensão dos paradigmas envolvidos na filosofia de programação orientada a objeto, é sugerida a leitura de Eckel (2000). O conceito de herança entre classes também é extensivamente utilizado no pacote NeoPZ.

### 2.2 A linguagem de programação C++

### 2.3 Exercício

#### 2.3.1 Ambientação

- Criar um projeto novo em XCode
- Rodar o programa principal que ele mesmo criou
- Acrescentar uma função que realize a operação  $f(x) = \sin(3\pi x)$
- Imprimir o resultado para  $0 \leq x \leq \pi$
- Executar o programa passo a passo utilizando o *debugger*
- Documentar um código utilizando Doxygen

### 2.3.2 Estudo prático da linguagem C++

#### Classe Vetor

Primeiramente, o aluno deve criar uma classe que implementa um vetor de números reais. No *header* da classe, os objetivos da classe, assim como os de quaisquer métodos que venham a ser criados, devem estar bem explicados. A classe criada deve conter, pelo menos:

- Construtor e destrutor
- Método que calcula a norma do vetor
- Métodos do tipo GetVal, SetVal
- Construtor de cópia
- Seleção de elemento através de operador [ ]
- Operador =
- Operador + e operador -
- Operador ==

*Obs: O livro Deitel and Deitel (2012) é fortemente recomendado para eventuais dúvidas, tanto relativas à sintaxe da linguagem quanto à ideia da classe em si*

#### Classe Matriz

Em seguida, o aluno deve criar uma classe que implementa uma matriz. Obviamente, alguns dos métodos criados no exercício anterior podem vir a ser aproveitados (diretamente ou não). Os atributos e métodos necessários para a classe devem ser pensados pelo aluno, sendo livre a forma da implementação e as estruturas de dado envolvidas. Entretanto, assim como na classe de vetor, alguns objetivos devem ser cumpridos:

- A matriz deve ter elementos do tipo *REAL* ( Ver *typedef* em Deitel and Deitel (2012))
- Construtor vazio;
- Construtor que cria uma matriz com  $m$  linhas e  $n$  colunas
- Construtor de cópia
- Destrutor
- Método para redimensionamento da matriz (*Resize*)
- Método que transpõe a matriz (*Transpose*)
- Métodos do tipo *GetVal* e *PutVal*
- Operador `()` para seleção de elemento da matriz
- Operador `=` para atribuição
- Operador `==` para comparação
- Operações de adição, subtração e multiplicação de matrizes (assim como seus respectivos operadores)
- Operação do produto entre matriz e vetor
- Uma função do tipo *debugStop()*

O aluno é incentivado a criar quaisquer métodos que venham a ser necessários para cumprir o objetivo. A classe deve ser testada com o auxílio de uma rotina que teste operações básicas possíveis de ser realizadas com ela. Ao final do exercício, o aluno deve responder as seguintes questões:

1. Qual o significado de *const* após a assinatura de um método?
2. Qual o significado de passar uma variável *const* por referência?
3. Quando são chamados os construtores e destrutores (*Dica: utilizando a função `std::cout`, faça que os construtores e o destrutor de sua classe imprimam uma mensagem quando forem chamados e o endereço do objeto responsável pela sua chamada*)?
4. Existe utilidade dos métodos *GetVal* e *PutVal* se há o operador parênteses para seleção de elemento?
5. Por que a utilização do *typedef double REAL*;



## 3 O Método dos Elementos Finitos (FEM): Uma introdução

### Introdução

Nesta atividade, o aluno deve estudar o primeiro capítulo de Oden et al. (1981) e realizar os exercícios propostos no decorrer do capítulo. Caso haja interesse em se aprofundar no assunto, a subseção 3.2 sugere a leitura do segundo capítulo do livro e realça os pontos importantes.

### 3.1 Oden - Capítulo I

No primeiro capítulo de Oden et al. (1981), o FEM é apresentado através da análise de um problema chamado problema modelo. O problema modelo se trata de uma equação diferencial com solução analítica conhecida, para fins de análise e validação. Claramente, métodos numéricos não justificam o esforço computacional em casos assim, entretanto o exemplo é posto para fins didáticos. A Formulação fraca, ou forma variacional, é apresentada como uma forma de relaxamento das condições necessárias para uma solução satisfatória. Ao invés de nos contentarmos única e somente com uma função  $u(x)$ , que satisfaça a equação diferencial em todos os pontos do domínio, buscamos agora uma função que satisfaça a nossa formulação fraca, ou seja, que satisfaça a nossa equação na forma integral. A nossa condição fraca também, naturalmente, diminui nossos critérios de suavidade quanto à solução desejada (na formulação variacional não aparece a segunda derivada da função  $u(x)$ ). É recomendada uma leitura atenta quanto aos espaços das funções  $u(x)$  e  $v(x)$ , dado que a discussão de qual é o espaço adequado para cada problema é de fundamental importância.

Neste capítulo também é abordada a técnica de aproximação de Galerkin, e como ela nos permite restringir o espaço de funções no qual buscamos nossa aproximação através da criação de um subespaço N-dimensional do espaço de funções analisado, caracterizado por um conjunto infinito das chamadas funções de base. Finalmente, o FEM é apresentado como a aplicação da técnica de aproximação de Galerkin sobre a forma fraca do nosso problema aliado a um método sistemático para a escolha das funções de base, sendo o domínio dividido em regiões - chamadas de elementos - e as funções de base definidas individualmente sobre cada elemento. Uma aproximação para a solução analítica é então apresentada, com o uso de funções chapéu e a divisão do domínio em quatro elementos. Os exercícios que acompanham o texto são altamente recomendados. Na seção 4 deste tutorial o aluno deve realizar um programa no software Mathematica que permite a visualização dos procedimentos realizados neste capítulo.

### 3.2 Opcional: Oden - Capítulo II

Caso, neste momento, haja um interesse grande em minúcias das aplicações do FEM em problemas unidimensionais é recomendada a leitura do segundo capítulo do livro. Neste capítulo temos uma abordagem mais geral de problemas unidimensionais, dando-nos a capacidade para utilização do método em qualquer equação diferencial linear de segunda ordem, assim como maior detalhamento na implementação de um algoritmo utilizando o FEM. Após a apresentação da chamada forma clássica de uma equação diferencial de segunda ordem, o texto segue para uma análise cuidadosa dos tipos de condições de contorno, e como estas condições aparecerão na formulação fraca é essencial para a elaboração de um bom algoritmo - Além, naturalmente, de uma maior compreensão do método. Também é abordada neste capítulo a questão da escolha das funções de base, i.e., quais as restrições para que um determinado conjunto de funções seja um conjunto adequado para a representação de um espaço. Com o uso de funções Lagrangianas, neste capítulo é desenvolvido um raciocínio que permite o uso de funções de base de maior ordem, ao invés de somente as funções chapéu.

Quanto à parte computacional, é apresentado em maiores detalhes o conceito de assemblagem, que permite que os cálculos sejam feitos individualmente em cada elemento e depois posicionados adequadamente em uma matriz global. Este conceito permite, por exemplo, a paralelização de tarefas em um determinado estágio de um algoritmo que apresenta uma implementação do FEM. Assim como no Capítulo 1, os exercícios deste capítulo também são recomendados. Na subseção 4.4 são sugeridas modificações no programa desenvolvido no Mathematica para fazer uso das técnicas apresentadas no Capítulo 2.

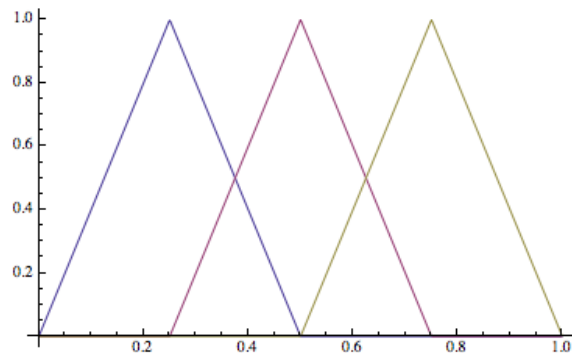
## 4 Introdução ao software Mathematica: Criando um primeiro código utilizando o FEM

### Implementação do Problema Modelo

Conforme abordado na subseção 3.1, neste tutorial do Mathematica a proposta é de criar um algoritmo para a resolução numérica do problema modelo. Para isto, é necessário ter em mente tanto a equação diferencial que origina a forma fraca como as condições de contorno do problema em questão. Assim como no Capítulo 1 de Oden et al. (1981), as funções de base escolhida serão lineares, as funções chapéu. Para melhor visualização da técnica, o algoritmo será feito de modo que o número de elementos em que o domínio será dividido seja variável, assim pode ser observada esta alteração no resultado final, quando comparado com a solução analítica.

#### 4.1 Implementação das funções chapéu

As funções chapéu são elaboradas conforme o procedimento seguido no Capítulo 1 de Oden et al. (1981), entretanto, desta vez criaremos um método geral para a divisão do domínio em qualquer número de elementos - todos com o mesmo comprimento. Assim sendo, como nosso domínio é a reta  $0 \leq x \leq 1$ , se tivermos  $nel$  elementos, cada um terá comprimento  $h = \frac{1}{nel}$ . Tendo em vista um nó interno  $x_i$  ao nosso domínio - i.e., excluindo os nós de fronteira, podemos dizer que a função de base correspondente será definida por  $\frac{x-(x_i-h)}{h}, x_i - h \leq x \leq x_i$  e  $\frac{x_i+h-x}{h}, x_i \leq x \leq x_i + h$ , tendo valor nulo para qualquer outro valor de  $x$ . Na Figura 2, vemos as funções de formas geradas ao dividirmos o domínio em 4 elementos (pode-se verificar que o resultado é idêntico ao apresentado em Oden et al. (1981)). O Mathematica apresenta também a função  $D[f,x]$ , que realiza a derivada de  $f$  em relação a  $x$ , e esta pode ser útil mais adiante no cálculo dos elementos da matrix  $K$  e do vetor  $F$ .



**Figura 2:** Funções de forma geradas pelo *software* Mathematica

## 4.2 Implementação das Estruturas de Dados

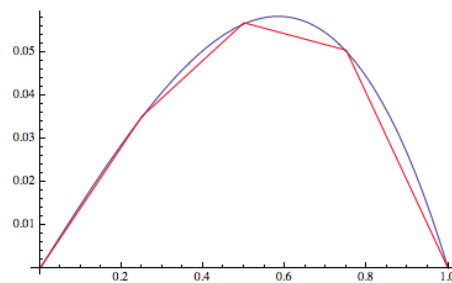
Nesta versão do algoritmo, tanto a Matriz de Rigidez quanto o Vetor de Carga serão computados de uma só vez, ao invés da soma das matrizes (ou vetores, no caso do vetor de carga) locais de cada elemento, apenas para fins de simplificação do algoritmo. Assim sendo, a função `Table` do Mathematica pode ser utilizada para criá-los. A função `MatrixForm` permite a visualização da matriz ou vetor passada como parâmetro na forma matricial, como pode ser visto na Figura 3.

$$\mathbf{K}: \begin{pmatrix} 8.16667 & -3.95833 & 0. \\ -3.95833 & 8.16667 & -3.95833 \\ 0. & -3.95833 & 8.16667 \end{pmatrix}$$
$$\mathbf{F}: \begin{pmatrix} 0.0625 \\ 0.125 \\ 0.1875 \end{pmatrix}$$

**Figura 3:** Matriz de Rigidez ( $K$ ) e Vetor de Carga ( $F$ ) gerados pelo algoritmo e visualizadas através do comando *MatrixForm*.

## 4.3 Resolução do Problema

Após obter a matriz  $K$  e o vetor  $F$ , o comando *LinearSolve* pode ser utilizado para resolver o sistema e encontrar os coeficientes (no caso de 4 elementos, confira-os com os coeficientes obtidos em Oden et al. (1981)). Lembrando que a aproximação é calculada a partir da somatória das multiplicações de  $\alpha_j \phi_j$ , o comando *Plot* pode ser utilizado para visualizar, por exemplo, a comparação da aproximação com a solução analítica do problema, como pode ser visto na figura 4.



**Figura 4:** Comparação da aproximação obtida com a solução analítica.

## 4.4 Opcional: Implementação de Funções Lagrangianas

Nesta seção o aluno deve implementar um algoritmo mais generalizado da resolução do problema modelo através do FEM. Baseado fortemente no algoritmo anterior, desta vez

alterações devem ser feitas de modo que a computação da matriz de rigidez e do vetor de carga seja feita separadamente para cada elemento - e depois seja implementado o processo de *assemblagem*, no qual cada matriz local é alocada na matriz global. Além disso, também é interessante a opção de que as funções de base a serem utilizadas desta vez sejam funções lagrangianas de ordem a ser especificada pelo usuário. Por mais que rotineiramente o uso de funções lagrangianas seja bastante diminuto comparado ao uso de funções hierárquicas, essas permitem ao aluno uma visão bastante clara da influência da ordem dos polinômios das funções de base.

#### 4.4.1 Elemento mestre

Conforme explicado no capítulo 2 do livro de Oden et al. (1981), todas as computações serão realizadas no chamado Elemento Mestre. Com o uso da coordenada generalizada  $\xi$ , cada elemento será mapeado de modo que, se o elemento  $i$  é delimitado pelos nós  $x_i$  e  $x_{i+1}$ , estes pontos serão mapeados, respectivamente, nos pontos  $\xi = -1$  e  $\xi = +1$ . As funções de forma serão definidas no domínio de  $\xi$ , e através da matriz de dependência, serão relacionadas às funções de base globais correspondentes (este processo será descrito em detalhes na subseção 4.4.3). Como deve estar claro neste ponto, as integrais também serão realizadas no domínio  $\xi = [-1, 1]$ , e portanto obteremos, por exemplo, para a matriz de rigidez local:

$$\begin{aligned} Kel_{j,k} &= \int_{x[i]}^{x[i+1]} [\phi_j(x)\phi_k(x) + \frac{d\phi_j(x)}{dx} \frac{d\phi_k(x)}{dx}] dx \\ &= \int_{-1}^1 [\psi_j(\xi)\psi_k(\xi) + \frac{d\psi_j(\xi)}{d\xi} \frac{d\xi}{dx} \frac{d\psi_k(\xi)}{d\xi} \frac{d\xi}{dx}] \frac{d\xi}{dx}^{-1} d\xi \end{aligned}$$

#### 4.4.2 Funções de Forma Lagrangianas

x As funções de forma utilizadas no algoritmo devem ser lagrangianas. Conforme definidas no Capítulo 2 de Oden et al. (1981), se deseja-se utilizar funções de ordem  $p$  e da subdivisão do elemento obtém-se  $p + 1$  pontos equidistantes  $x_i, i = 1, 2, \dots, p + 1$  (que são mapeados no domínio  $\xi$ ) as funções de forma são, portanto:

$$\psi_i(\xi) = \prod_{j=1, j \neq i}^{p+1} \frac{\xi - \xi_j}{\xi_i - \xi_j}$$

Com a propriedade de:

$$\psi_i(\xi_j) = \begin{cases} 1, & j = i \\ 0, & j \neq i \end{cases}$$

#### 4.4.3 Assemblagem

Assemblagem é o processo de posicionar corretamente as matrizes locais geradas na computação individual de cada elemento na matriz global. Isso equivale a associar cada função de forma (definidas no elemento mestre, portanto, em  $\xi$ ) à função de base equivalente (definida no domínio do problema, portanto, em  $x$ ). Assim, se temos  $nel$  elementos e  $p + 1$  funções de forma, nossa matriz de dependência terá  $nel$  linhas e  $p + 1$  colunas, sendo o elemento  $[dependencyMatrix]_{i,j}$  a função de base associada à  $j$ -ésima função de forma do

elemento  $i$ . No caso de funções lagrangianas, é bom lembrar que a primeira função de forma do elemento  $i + 1$  corresponde à mesma função de base do que a última função de forma do elemento  $i$ , portanto, elas devem ter o mesmo coeficiente  $\alpha_j$  associado a elas, *i.e.*, o mesmo grau de liberdade.

Matriz de dependencia:

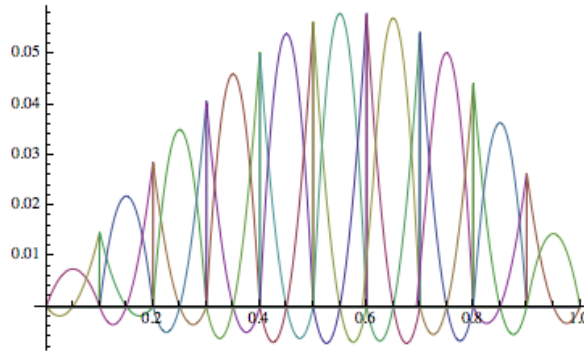
1	2	3
3	4	5
5	6	7
7	8	9
9	10	11
11	12	13
13	14	15
15	16	17
17	18	19
19	20	21

**Figura 5:** Matriz de dependência gerada no caso da divisão do domínio em dez elementos e funções de forma de segundo grau.

#### 4.4.4 Condições de Contorno

No caso de condições de contorno de Dirichlet (Ver seção 2.7.4 de Oden et al. (1981)), queremos forçar, por exemplo,  $\alpha_1 = v_1$ . Temos  $\alpha_1 K_{1,1} + \alpha_2 K_{1,2} + \dots + \alpha_N K_{1,N} = F_1$  (onde  $N = nel * p + 1$ ), portanto, fazendo  $F_1 = v_1 * BIG$  e  $K_{1,1} = BIG$ , onde  $BIG \gg 1$  forçamos  $\alpha_1 = \frac{F_1 - (\alpha_2 K_{1,2} + \dots + \alpha_N K_{1,N})}{K_{1,1}} = v_1$ . O mesmo raciocínio pode ser utilizado para impor um valor para  $\alpha_N$ .

Funcoes de base multiplicadas pelos seus respectivos coeficientes:



**Figura 6:** Funções de base multiplicadas pelos seus respectivos coeficientes, gerados no caso da divisão do domínio em dez elementos e funções de forma de segundo grau. A aproximação obtida corresponde à soma destas funções.

## 5 O pacote NeoPZ e a implementação do Problema Modelo

Nesta seção o aluno deve instalar o Pacote NeoPZ, rodar uma simulação já implementada e analisar seu algoritmo, visando a ambientação com o pacote.

### 5.1 Dependências

Para a instalação do pacote NeoPZ, siga o tutorial encontrado no site do LabMeC em <http://www.labmec.org.br/wiki/howto/start>.

### 5.2 Exemplo Simples

Após a instalação do pacote NeoPZ o aluno deve executar o programa *ExemploSimples.cpp*, no qual é implementado o Problema Modelo do primeiro capítulo de Oden et al. (1981) com número variável de elementos. O aluno deve então ler o código, entendendo cada linha deste, para se ambientar com o ambiente NeoPZ - inclusive o material criado para este exemplo, *TPZMatModelProblem*, que implementa a formulação fraca do problema modelo. Entretanto, um aviso: Não é esperado que o aluno compreenda todas as estruturas de dados utilizadas no NeoPZ para a resolução de um problema, somente a estrutura de dados de forma geral, apresentada nos arquivos *ExemploSimples.cpp*, *TPZMatModelProblem.h* e *TPZMatModelProblem.cpp*. Após a execução do programa, será gerado um arquivo *.vtk* que poderá ser visualizado utilizando o software *Paraview*, e também será gerada no próprio console de execução a saída na formatação compatível com o Mathematica, para fins de comparação. É esperado que o aluno verifique se foram obtidos os mesmos resultados obtidos na seção 4.

## 6 Bibliografia

Paul Deitel and Harvey Deitel. *C++ How to Program*. Prentice-Hall, 8 edition, 2012.

Bruce Eckel. *Thinking in C++: Introduction to Standard C++*, volume 1. Prentice Hall, 2, 2000.

LabMeC. Laboratório de Mecânica Computacional. URL <http://www.labmec.org.br>.

J. T. Oden, E. B. Becker, and G. F. Carey. *Finite Elements: An Introduction Volume I*. Prentice-Hall, 1981.



**A   Phillipe R. B. Devloo - Simulação Numérica**

## SIMULAÇÃO NUMÉRICA

*Philippe R. B. Devloo<sup>(A)</sup>*

### INTRODUÇÃO

Escrever um texto sobre simulação numérica para um público não especializado na área é um grande desafio. Na composição do texto fui obrigado a refletir sobre o tema que estou pesquisando de forma diferente da que tenho costume. Tentei escrever de forma que o conteúdo seja interessante para o público técnico e não técnico. O texto é uma reflexão sobre a área baseado em minha experiência profissional.

Na primeira seção faço uma reflexão sobre o conceito de simulação. A simulação como conceito existe desde a existência do homem civilizado. A simulação executada por computador existe desde o advento do próprio computador. Na segunda seção enfatizo a importância da simulação numérica para a competitividade industrial. Caso o país não desenvolva sua competência em simulação numérica, ele estará fadado a ser um pólo produtor de projetos importados. Na terceira seção tento elencar vários tipos de simulação numérica. Destaco a área de mecânica computacional pela sua sistemática de desenvolvimento e pela sua abrangência de aplicações. Demonstro como a área de mecânica computacional é multidisciplinar e quais áreas ela envolve. Delineio o perfil do profissional da área.

Comenta-se a presença da área de mecânica computacional na UNICAMP e a atuação do LabMeC (Laboratório de Mecânica Computacional) da Faculdade de Engenharia Civil. Finalmente, destaco as dificuldades enfrentadas pelos profissionais da área, tanto a nível institucional quanto ao seu relacionamento com os órgãos de fomento à pesquisa.

### SIMULAÇÃO NUMÉRICA

A simulação faz parte da capacidade humana de imaginação. O nosso cérebro cria constantemente imagens da realidade à sua volta. Platão até questiona a validade da sua

visão do mundo: será que o que nós vemos é real ou será que nós vemos apenas uma sombra da realidade projetada numa parede. Imaginação e simulação são conceitos correlatos: na imaginação criamos um sistema no qual a relação causa-efeito é imaginada e comparada com a realidade. Uma das nossas capacidades mais refinadas de simulação é a intuição: nela, um modelo refinado da realidade é criado na mente, na qual a relação causa-efeito é prevista, sem por isso ter um modelo racional do fenômeno e nem uma visão clara da causa.

Uma das áreas mais desenvolvidas de simulação por computador é a criação de ambientes virtuais. Exemplo dessa área de simulação são os jogos de computador: neles, é criado um mundo virtual no qual o jogador interage. À medida que a potência dos computadores aumenta, o realismo desses ambientes torna-se cada vez mais convincente. O impacto desses simuladores sobre o mundo da computação é tamanho que o jogo de computador é considerado o maior motivo econômico para o desenvolvimento de computadores pessoais mais potentes. Esta área não é a área de atuação do autor.

Outra área é a simulação por computador de padrões de tomadas de decisão encontrados na natureza. Esses incluem a simulação de mecanismos de seleção e/ou padrões de raciocínio praticados pelo cérebro. A área de algoritmos genéticos ganhou impulso pelo trabalho de John Nash, cuja biografia foi representada no filme "Uma Mente Brilhante". Esta área tem sido de pesquisa muito ativa. As suas áreas de aplicação incluem otimização multiobjetivo, *data mining*, etc. As redes neurais são uma representação em estruturas de computador das ligações entre neurônios. Redes neurais conseguem prever respostas em função de um conjunto de parâmetros de entrada, mediante um treinamento de um conjunto de valores entrada/saída.

Finalmente, a área de mecânica computacional procura simular diversos fenômenos físicos utilizando uma sistemática que envolve engenharia, matemática e ciência da computação. O fenômeno físico em estudo é representado por um sistema de equações parciais diferenciais (tradução do problema de engenharia para a matemática). O sistema de equações é aproximado pelo método de elementos finitos (problema de matemática para

problema de matemática aplicada e de ciência da computação). Finalmente os resultados da simulação são comparados com o fenômeno físico em estudo (engenharia).

Os fenômenos físicos podem ser os mais variados, indo da simulação estrutural à simulação de fluídos e gases, ou até mesmo circuitos elétricos.<sup>(4)(5)</sup> Em certas áreas a simulação por elementos finitos é consagrada; em outras, as técnicas ainda estão em fase de desenvolvimento.

A área de mecânica computacional tem ganho impulso pela disponibilidade de computadores potentes a preços acessíveis. Qualquer computador de mesa hoje tem o poder computacional de um supercomputador de 20 anos atrás.<sup>(B)</sup> A simulação numérica está ao alcance de qualquer empresa, desde que tenha o software e os recursos humanos adequados.

Os recursos humanos necessários para o desenvolvimento da área de mecânica computacional são diversos: precisa-se de pesquisadores para o desenvolvimento de modelos matemáticos para fenômenos físicos, de pesquisadores para aproximar esses modelos com algoritmos de elementos finitos, de pesquisadores para o desenvolvimento de softwares e de engenheiros para utilizar esses softwares e tirar conclusões adequadas. Trata-se, por motivos óbvios, de uma área multidisciplinar de pesquisa, na qual engenheiros de todas as áreas de atuação, matemáticos e cientistas de computação colaboram em uma área comum.

## **O SIGNIFICADO DA SIMULAÇÃO NUMÉRICA PARA A COMPETITIVIDADE INDUSTRIAL**

O modelo de produção industrial atual favorece uma produção em massa quase totalmente automatizada. Nela o produto é realizado através de uma série de manipulações por maquinários. A origem deste processo produtivo é freqüentemente uma evolução de um processo fragmentado que foi gradualmente automatizado. Isto é, chegou-se a automatização da produção passo a passo, num demorado processo de aferições e ajustes.

Vimos por outro lado uma grande pressão pelo público consumidor, para o oferecimento de produtos customizados que aparentemente não são produzidos em massa (mas com preço somente alcançável em produção automática). Um exemplo desse tipo de customização é a opção oferecida pela empresa Nike para a compra de tênis totalmente customizados pela internet, a serem entregues em casa pelo correio.

Este argumento demonstra a origem da demanda sobre o processo produtivo ser não somente por preço competitivo (i.e. automatizado) mas também pela necessidade de saber adaptar/customizar a produção em tempo menor que o praticado atualmente.

A simulação numérica do processo produtivo tem um impacto muito grande sobre o mesmo:

- Para realizar a simulação deve-se conhecer o processo de produção profundamente. Este estudo e/ou questionamento do processo normalmente já resulta em melhorias imediatas.
- Tendo a simulação da produção, é possível modificá-la virtualmente (sem interferir no processo funcional) e otimizá-la de acordo com um dado objetivo (e.g. aumento de produção e/ou redução de custos).
- Novos produtos podem ser desenvolvidos em ambiente virtual, diminuindo o seu tempo de desenvolvimento. O produto que entra em produção requer menos aferição. Um exemplo clássico deste impacto é a indústria automobilística. O tempo entre concepção de um carro e sua produção foi reduzido de aproximadamente três anos para um ano e meio.<sup>(C)</sup> A montadora consegue trazer inovações tecnológicas mais rapidamente ao mercado.

Pode parecer simples e que apresentemos a cura milagrosa para automação: basta simular para melhorar. De fato, desenvolver uma simulação é um processo custoso que requer pessoas especializadas e qualificadas. Esta publicação quer demonstrar os benefícios

da implantação do processo de simulação. O custo da sua implantação deve ser avaliado caso a caso.

## **A ÁREA DE MECÂNICA COMPUTACIONAL**

A área de mecânica computacional apresenta uma forma sistemática para desenvolver processos de simulação numérica. Este processo está sendo praticado em todas as áreas de engenharia, na matemática aplicada e na ciência da computação. Muitos processos podem ser simulados com softwares comerciais disponíveis, mas mesmo nas áreas na qual a simulação numérica é amplamente utilizada, existem tópicos de pesquisa ativos, tais como otimização, controle de qualidade de aproximação, otimização de tempo de execução, etc.

## **SISTEMÁTICA DE SIMULAÇÃO**

A maioria dos fenômenos físicos respeita uma ou mais leis de conservação. Para a engenharia estrutural, a conservação de movimento linear é fundamental. Para a mecânica dos fluidos acrescenta-se a conservação de massa e energia. Em outras áreas de engenharia, outras quantidades são conservadas.

Em um segundo passo a cinemática do fenômeno é estudada: a cinemática de sólidos para a engenharia estrutural e/ou a cinemática de partículas de fluido para a mecânica dos fluidos.

Finalmente a lei constitutiva relaciona a cinemática e a quantidade conservada. Chega-se, desta maneira, a um sistema de equações diferenciais que aproximam o fenômeno estudado.

Quase todos os modelos envolvem uma simplificação do objeto físico: o modelo adequado é aquele que pode ser aproximado por técnica numérica em tempo hábil e que consegue capturar a essência do fenômeno em estudo. O aprimoramento da simulação é um tema ativo de pesquisa.

## ÁREAS ENVOLVIDAS

A prática da mecânica computacional é uma atividade multidisciplinar.

- A área de engenharia apresenta o problema a ser estudado. A mecânica computacional está presente em todas as áreas de engenharia. Neste sentido ela transcende a diferença que atualmente existe entre as várias especialidades de engenharia. Na faculdade de Engenharia Civil da UNICAMP encontram-se professores dedicados à simulação numérica de algum fenômeno em todos os departamentos.
- A matemática tem tido um impacto decisivo sobre a mecânica computacional. A análise funcional é a ferramenta fundamental para provar a consistência e convergência do método de elementos finitos.<sup>(D)</sup> Sem a garantia de convergência e qualidade da aproximação numérica, os engenheiros não poderiam confiar nos resultados obtidos pelos métodos de aproximação. O estudo de métodos eficientes para resolução de grandes sistemas de equações tem permitido simular problemas com um realismo cada vez maior. Finalmente, a teoria de otimização aplicada à aproximação numérica, juntamente com uma função objetiva, permite otimizar o objeto em estudo.
- Qualquer simulação numérica passa pelo desenvolvimento de um programa de computador. Esses programas precisam ser eficientes e adaptados à arquitetura do computador. Grandes avanços têm sido feitos na última década em desenvolvimento de linguagens de programação de alto nível de abstração, porém eficientes. A implementação dos programas em computadores paralelos é ainda um desafio.

## PERFIL DO PROFISSIONAL

Na aplicação da simulação numérica existem basicamente dois perfis de profissionais: os versados na utilização de programas comerciais de simulação para analisar projetos e os desenvolvedores de programas de simulação numérica.

O profissional com formação em simulação numérica deve focar o objeto a ser estudado, com ênfase à análise do modelo desenvolvido e à interpretação dos resultados. O profissional com foco no desenvolvimento de algoritmos tem uma visão mais teórica do processo de simulação.

Formar um aluno de pós-graduação na área de mecânica computacional é um desafio. Qualquer que seja a sua formação na graduação, o aluno deve receber uma formação complementar nas áreas que não cursou. Isto entra em conflito com a pressão da pós-graduação em formar alunos num prazo cada vez menor.

No departamento de estruturas da Faculdade de Engenharia Civil, o aluno ingressante cursa uma disciplina na área de mecânica dos sólidos, na qual aprende os fundamentos da mecânica dos meios contínuos. Ele cursa também uma disciplina introdutória de análise estrutural, na qual é exigido o desenvolvimento de um programa de análise estrutural. No desenvolvimento do seu tema de pesquisa, o aluno ainda irá aprofundar-se num tópico especializado da mecânica computacional.

## **A MECÂNICA COMPUTACIONAL NA UNICAMP**

A mecânica computacional está presente em todas as áreas tecnológicas da UNICAMP. É comum dizer que hoje quase tudo se faz por computador. Certamente a engenharia sente esse impacto da área de simulação numérica cada vez mais fortemente. Todo ano abrem-se novas áreas de aplicação para a simulação numérica:

- Na medicina desenvolveu-se a área de simulação de escoamento de sangue nas veias (hemodinâmica). Este estudo é significativo no projeto das cirurgias de ponte de safena. Com simulação numérica é possível estudar a eficiência do escoamento sangüíneo após a cirurgia.
- Na área de odontologia estuda-se a concentração de tensões provocada por implantes dentários. A simulação numérica pode nos ajudar a escolher os materiais mais apropriados para cada implante.



- O comportamento estrutural de barragens, edifícios, pontes e outras construções civis sempre é analisado com softwares de simulação, antes da construção.
- O campo eletromagnético provocado por antenas pode ser simulado em três dimensões. O possível aquecimento das células cerebrais pelo uso freqüente do telefone celular tem sido um tópico muito comentado.

Em todas as áreas mencionadas existem profissionais na UNICAMP dedicados à simulação numérica. Todos enfrentam os mesmos problemas de formação dos alunos e da combinação de engenharia, matemática e ciência da computação. Se fosse possível criar uma entidade dedicada à mecânica computacional, esta mesma poderia prestar serviços de formação de alunos. Esta organização irá diminuir a sobreposição existente e melhorar a qualidade do ensino.

## **O LABORATÓRIO LABMEC**

O objetivo do Laboratório de Mecânica Computacional é agregar os professores pesquisadores do Departamento de Estruturas da Faculdade de Engenharia Civil que atuam na área de mecânica computacional. Como tal, ele deve atuar como pólo multidisciplinar para o desenvolvimento de simulação numérica. O LabMeC possui um parque de computadores de última geração, interligados em rede de alta velocidade e configurados com o sistema Linux (<http://labmec.fec.unicamp.br>).

Diversos projetos de pesquisa já foram desenvolvidos no LabMeC com apoio de órgãos de fomento como a Fapesp, CNPq e projetos de cooperação com a indústria. Dois projetos no LabMeC dizem respeito ao desenvolvimento de softwares de domínio público, e por isso merecem destaque particular

**PZ**

O ambiente PZ é um ambiente escrito em linguagem de computador, orientada para objetos, para o desenvolvimento de algoritmos de elementos finitos.<sup>(1)</sup>

O desenvolvimento de algoritmos sofisticados para o método de elementos finitos requer um investimento muito grande da parte do pesquisador. Isto inviabiliza o envolvimento de pesquisadores em diversas áreas, porque durante o desenvolvimento deste software eles não têm material para publicação.

Por exemplo, demoramos no LabMeC três anos para implementar o refinamento hp adaptativo tridimensional<sup>(E)</sup> para o método de elementos finitos. Isto considerando que o autor desenvolveu sua tese de doutorado no tema. Qual outro pesquisador poderá investir este tempo no desenvolvimento de uma técnica que já foi implementada? O ambiente incorpora diversas técnicas que, por serem implementadas em linguagem orientada por objetos, podem ser reaproveitadas.

## **OOPar**

O limite dos problemas que podem ser simulados sempre será a potência do computador do qual dispomos (e a eficiência dos simuladores). Recentemente os construtores de processadores admitirem que encontrarão uma barreira de frequência de relógio dos mesmos (não haverá pentium de 5 GHz). A solução encontrada é dotar os computadores de várias processadores que podem dividir as tarefas da máquina. Este tipo de computador é chamado de computador multiprocessado.

Um dos maiores desafios em desenvolvimento de softwares é a adaptação de programas para computadores multiprocessados. O projeto OOPar oferece um ambiente de programação para o desenvolvimento de programas paralelos.<sup>(F)</sup>

O OOPar apresenta uma abordagem de desenvolvimento de programas paralelos inovadora, introduzindo o conceito de dados distribuídos e tarefas que atuam sobre os dados. O OOPar não introduz uma linguagem de computador estendida. Ele define um

abordagem para computação paralela e implementa uma biblioteca com interface simplificada.<sup>(3)(2)</sup> O código é aberto e disponível para qualquer um.

Seguindo a filosofia do projeto PZ, o OOPar proporciona ao pesquisador uma ferramenta que simplifica a criação de programas paralelos sem restrições ao seu uso.

## O DESENVOLVIMENTO DA ÁREA

**(UNICAMP)** Como mencionado anteriormente, a área de mecânica computacional está presente na maioria das unidades da UNICAMP. Por outro lado, não há nenhuma política institucional para o desenvolvimento da área. Neste ponto, cada unidade desenvolve e oferece seus cursos de elementos finitos, programação e álgebra linear. Naturalmente, é possível cursar disciplinas de pós-graduação em outras unidades, mas o canal de comunicação não está estabelecido.

Utilizamos este canal de comunicação para pleitear a formação de um centro ou núcleo dedicado à área de simulação numérica/mecânica computacional. A missão de tal centro poderia estar nas áreas de:

- Formação de alunos
  - Oferecimento de cursos de pós-graduação dedicados à simulação numérica. Estes cursos seriam das áreas de engenharia, matemática aplicada e ciência da computação.
  - Oferecimento de cursos optativos para alunos de graduação e coordenação de projetos de iniciação científica.
- Centro multidisciplinar para professores
  - O centro poderá ser o canal de comunicação para os docentes que atuam na área. A troca de idéias entre profissionais de diferentes áreas, mas que se dedicam ao mesmo tema, será muito enriquecedor.
- Prestador de serviços

- O centro poderá agrupar a capacitação existente em simulação numérica e apresentar a UNICAMP como centro de excelência em simulação numérica.

**(Brasil)** A área de mecânica computacional não aparece na classificação das áreas de fomento à pesquisa. As áreas seguem as organizações tradicionais das universidades e não contemplam áreas multidisciplinares. Sendo assim, pesquisadores atuando em simulação numérica submetem projetos nas áreas mais variadas.

Criando uma área específica para simulação numérica pode-se alcançar os seguintes benefícios:

- Avaliação de projetos por pesquisadores dedicados à área. Na situação atual, um projeto submetido por um pesquisador que atua na área de elementos finitos não tem garantia de que seu projeto será julgado por alguém da sua área.
- Não existe orçamento para a área de mecânica computacional. O investimento para a área de análise está espalhado em todas as áreas de fomento. Não há visão clara de quanto esta área absorve de investimento em pesquisa e/ou quantos pesquisadores estão sendo formados na mesma.
- Considerando que a área não existe na classificação CAPES, não existem projetos temáticos dirigidos para a mesma.

## CONCLUSÕES

Tentamos elencar a área de simulação numérica com ênfase na área de mecânica computacional. Com a disponibilidade de computadores potentes a preços acessíveis, a simulação numérica está ao alcance de todos. O impacto da área sobre o desenvolvimento industrial do país é muito grande. A competitividade em produção industrial depende da capacidade de simular os processos existentes e dos processos futuros.

Muitos pesquisadores dedicam-se à área de simulação numérica nas instituições de ensino e pesquisa, mas eles estão espalhados nas diversas repartições das mesmas. As estruturas das mesmas favorece a especialização em detrimento da multidisciplinaridade.

O autor é pesquisador na área de elementos finitos na Faculdade de Engenharia Civil da UNICAMP. Sua pesquisa inclui dois projetos cujos objetivos são disponibilizar ambientes de desenvolvimento de softwares de elementos finitos e computação paralela para a comunidade acadêmica. Caso esses softwares sejam adotados por diferentes equipes, abrir-se-á um caminho para o diálogo multidisciplinar em torno da mecânica computacional.

As instituições de pesquisa não estão respondendo adequadamente às necessidades de uma área multidisciplinar tal como a área de mecânica computacional. A qualidade de pesquisa e o desenvolvimento da área de mecânica computacional seriam beneficiados com a atuação explícita das instituições. Na situação atual há uma sobreposição grande de temas pesquisados. Há também uma sobreposição desnecessária de cursos básicos na área.

## **AGRADECIMENTOS**

Agradeço os alunos do LabMeC pelo ambiente estimulante de trabalho nestes últimos anos. Agradeço em particular o Eng. Edimar Cesar Rylo pela revisão do texto. Agradeço minha esposa pela revisão e pelas discussões sobre o mesmo.

## **NOTAS**

- (A) Faculdade de Engenharia Civil – UNICAMP; [phil@fed.unicamp.br](mailto:phil@fed.unicamp.br) (abril de 2005).
- (B) Nestes computadores foram simulados os projetos do Boeing 747 e do Space Shuttle.
- (C) Este tempo foi reduzido devido à simulação mas também devido à tecnologia da informação em geral.
- (D) O método de elementos finitos é o método mais utilizado para aproximar sistemas de equações parciais diferenciais.
- (E) Adaptividade tridimensional hp diz respeito a técnicas de controle de erro de aproximação.

(F) Um programa paralelo é um programa adaptado para um computador multiprocessado.

### REFERÊNCIAS BIBLIOGRÁFICAS:

- (1) P. R. B. Devloo. PZ: An object oriented environment for scientific programming. *Computer Methods in Applied Mechanics and Engineering*, 150:133--153, 1997.
- (2) P. R. B. Devloo. Object oriented programming applied to the development of scientific software. In E.Dvorkin S.Idelsohn, E.Oñate, editor, *Computational Mechanics, New Trends and Applications*, Edificio C-1 Campus Nord UPC, Gran Capità, s/n, 08034 Barcelona, Spain, June 1998. Fourth World Congress on Computational Mechanics (IV WCCM), CIMNE.
- (3) P. R. B. Devloo, F. A. M. Menezes, and E. C. Silva. OOPAR: The development of an environment for parallel computing using the object oriented programming philosophy. In B.H.V. Topping, editor, *Advances in Computational Structures Technology*, pages 151-156, 10 Saxe-Coburg Place, Edinburgh, EH3 5BR, UK, 1996. CIVIL-COMP Press.
- (4) Hayrettin Kardestunder and Douglas H. Norrie. *FINITE ELEMENT HANDBOOK*. McGraw-Hill, 1987.
- (5) O. C. Zienkiewicz. *The Finite Element Method in Engineering Science*. McGraw-Hill Publishing Company Limited, Maidenhead, 1971.