Asociación Argentina de Mecánica Computacional

Mecánica Computacional Vol XXIX, págs. 4721-4732 (artículo completo)
Eduardo Dvorkin, Marcela Goldschmit, Mario Storti (Eds.)
Buenos Aires, Argentina, 15-18 Noviembre 2010

CONTRIBUIÇÕES TEÓRICAS PARA INCORPORAÇÃO DO MÉTODO ELEMENT-FREE GALERKIN EM SOFTWARES BASEADOS NO MÉTODO DOS ELEMENTOS FINITOS

Gleber N. Marques[a,b] **e Regis B. Perez**[a]

[a]*Grupo Laboratório de Modelagem e Simulação Computacionalde Alto Desempenho (LAMSCAD), Universidade do Estado de Mato Grosso (UNEMAT), Alto Araguaia 78780-000, Mato Grosso, Brasil, gleber.nmarques@gmail.com, regisbatistaperez@gmail.com http://dgp.cnpq.br/buscaoperacional/detalhegrupo.jsp?grupo=8415103BIS3DKU*

[b]*Departamento de Computação, Universidade do Estado de Mato Grosso, Campus Universitário de Alto Araguaia, Alto Araguaia 78780-000, Mato Grosso, Brasil, http://www.aia.unemat.br*

Palavras Chave: métodos meshfree, EFGM, MEF, engenharia de software, CAD/CAE.

Resumo. Modelos matemáticos cada vez mais complexos vêm sendo propostos e utilizados para modelar problemas de interesse nas engenharias e ciências, resolvidos por meio de métodos computacionais adequadamente escolhidos. Um método relativamente recente que vem continuamente conquistando espaço em softwares de simulação em engenharia é o método meshfree Element-Free Galerkin (EFGM). Em contrapartida o Método dos Elementos Finitos (MEF) é utilizado há bastante tempo, sendo que muitos softwares de simulação computacional foram desenvolvidos implementando o MEF. O EFGM tem apresentado bons resultados em diversos campos e em particular representa uma alternativa robusta nos casos que uma aproximação usando malha não é viável. Assim, é natural o interesse e relevância do estudo e disseminação de modelos computacionais que permitam estender as funcionalidades de um software CAD/CAE (*Computer-Aided Design/Computer-Aided Engineering*) que implementa o MEF para que seja possível oferecer também um *solver* EFGM. Neste contexto, o Instituto de Estudos Avançados do Centro Técnico Aeroespacial (IEAv/CTA), em colaboração com outros pesquisadores, coordena o desenvolvimento da plataforma Levsoft, um software orientado a objetos em linguagem C++ para simulação de problemas de engenharia usando MEF. O estudo e análise dos fundamentos teóricos relacionados com a aplicação do EFGM na resolução da equação de Poisson e nas semelhanças com a implementação MEF implicou num estudo de caso que resulta na proposta de um modelo computacional para uma arquitetura de classes que incorpora o EFGM ao Levsoft, buscando-se preservar a coerência conceitual entre as classes que implementam ambos os métodos além de uniformidade do código Orientado a Objetos (OO).

1 INTRODUÇÃO

Nas últimas décadas o hardware deixou de ser o item mais caro na implementação de um sistema, enquanto o custo relacionado ao software cresceu e se tornou o principal item no orçamento de implantação de sistemas computacionais (Pressman, 1995). Isso se deve principalmente à crescente complexidade dos problemas que hoje podem ser resolvidos computacionalmente, o que se tornou possível graças à crescente disponibilidade de capacidade de processamento e memória a baixos custos aliado ao constante desenvolvimento de linguagens e ferramentas de programação e, em alguns casos, graças à ampliação do acesso a arquiteturas de processamento de alto desempenho tipo clusters.

Com a disponibilidade de alta capacidade de processamento e memória, em especial, a complexidade dos sistemas de simulação em Engenharia, doravante Computer Aided Design / Computer Aided Engineering (CAD/CAE), cresceu em resposta à demanda crescente pelo estudo de novas tecnologias, dispositivos e pela busca de projetos otimizados, de modo que esses sistemas deixaram de ser uma ferramenta exclusiva para engenheiros e vêm se tornando cada vez mais necessários em diversos segmentos científicos e tecnológicos.

Por muito tempo a capacidade de processamento e de memória limitou a aplicação e uso de determinados métodos numéricos, inclusive desencorajavam sua própria investigação devido ao custo computacional envolvido. A partir do início da superação dessas restrições de desempenho que ocorreu em torno do final da década de oitenta em alguns países, observa-se na literatura especializada um aumento significativo do número de métodos e modelos computacionais propostos envolvendo custos computacionais mais elevados que os tradicionais métodos dos elementos finitos (MEF) e de Diferenças Finitas (DF), como os métodos meshfree/meshless (Nayroles et al., 1992; Belytschko et al., 1994).

Naturalmente a principal motivação ao estudo dos métodos meshless residiu inicialmente nas limitações de aplicação dos métodos baseados em malha impostas pelo processo de geração de malha, cuja regularidade e qualidade interferem diretamente na acurácia da aproximação fornecida por esses métodos. Assim, o principal interesse no estudo desta recente classe de métodos em especial do EFGM (Belytschko et al., 1994) é que o cálculo das funções de base (ou funções de forma), a partir das quais a aproximação é construída, não depende de uma malha minimamente adequada, como ocorre no método dos elementos finitos (FEM), mas depende somente de um conjunto de nós provenientes da discretização do domínio e de funções peso associadas a cada nó. A conectividade entre os nós é estabelecida somente pela avaliação destas funções peso, e nenhuma informação específica sobre a malha é requerida para o cálculo dos valores das funções de forma. Estas características permitem aos métodos sem malha superar a maioria dos obstáculos comuns das técnicas numéricas baseadas em malha. Por exemplo, a degradação da acurácia da aproximação quando elementos distorcidos surgem, e o alto custo computacional do "remalhamento" do domínio durante o processo de simulação, em particular para geometrias complexas 3D.

Em particular, o EFGM foi desenvolvido por Nayroles et al. (1992) e aprimorado por Belytschko et al. (1994) para cálculos de estruturas na engenharia mecânica e civil. Rapidamente, sua utilização se estendeu à engenharia elétrica, magneto-hidrodinâmica, física de plasmas e física quântica dentre outras (Marques et al., 2007), apresentando bons resultados numéricos, muito embora, em geral, apresente maior custo computacional comparado ao MEF. Contudo, conforme discutido em Marques et al. (2007), Xuan et al. (2004) e Belytschko e Krysl (2001), uma comparação de desempenho computacional rigorosa deve levar em conta o grau de convergência das aproximações além do tempo de processamento, tendo em vista que o EFGM apresenta convergência *h* mais rápida que o MEF o que frequentemente não é considerado (Marques et al., 2007).

O potencial sucesso de aplicação do EFGM para estudo dos mais variados dispositivos das engenharias e ciências torna-o um candidato promissor para a implementação de *solvers* (núcleos de cálculos) presentes em softwares CAD/CAE, adicionalmente ao MEF, tendo em vista o interesse em ampliar as possibilidades de cálculo desses sistemas usufruindo dos módulos de desenho (CAD) e de visualização pré-existentes. (Seidl e Schmidt, 2006; 2007; Frauenfelder e Lage, 2002)

Essa necessidade cada vez mais freqüente da construção de softwares de alta complexidade que disponham de variados algoritmos e implementações alternativas para serem usados como ferramentas de interesse para engenheiros e cientistas em campos variados exige uma abordagem com visão de engenharia de software consistente no processo de desenvolvimento para viabilizar o aprimoramento do sistema (Seidl e Schmidt, 2006; Frauenfelder e Lage, 2002). A seqüência de desenvolvimento, dentro do paradigma de orientação a objetos, deve basear-se na análise conceitual exaustiva, no projeto detalhado e documentado, e numa programação baseada em modularidade e reutilização de componentes de softwares previamente construídos e testados que viabilize a permanente manutenção do sistema no que concerne a integração de novos módulos ou funcionalidades, consonante com o objetivo de atender à demanda pela incorporação de novos métodos numéricos a sistemas CAD/CAE já disponíveis.

O presente trabalho descreve como podem ser exploradas algumas relações conceituais existentes entre os métodos EFGM e MEF em um processo de engenharia de software Orientado a Objetos (OO) empregado na análise de um protótipo construído para incorporação do EFGM em um sistema CAD/CAE que implementa o MEF denominado LEVSOFT (Paes et al., 2002). São discutidas as principais vantagens que podem ser exploradas para a implementação de um *solver* EFGM em um sistema CAD/CAE baseado em MEF, bem como contextualizamos algumas situações limite para uma aplicação MEF e o correspondente impacto no caso de uma aplicação do EFGM.

2 MODELOS MATEMÁTICOS

2.1 O Método dos Resíduos Ponderados e o Procedimento de Galerkin

Considere o seguinte problema de valor de contorno correspondente à equação de Poisson em meio isotrópico para o potencial escalar elétrico $\Phi(\mathbf{x})$:

$$\nabla^2\Phi(\mathbf{x}) = \rho\,(\mathbf{x}), \tag{1}$$

$$\Phi(\mathbf{x}) = \phi_{Dir}, \qquad em\ \Gamma_{Dir} \in \Gamma \tag{2}$$

$$\frac{\partial\Phi}{\partial n} = f_{Neu}, \qquad em\ \Gamma_{Neu} \in \Gamma \tag{3}$$

onde, $\Gamma_{Dir} \cap \Gamma_{Neu} = \{\}$, $\Gamma_{Dir} \cup \Gamma_{Neu} = \Gamma$ é o contorno do domínio e ρ é uma função escalar conhecida ao menos sobre os pontos de discretização denominados pontos nodais ou simplesmente nós.

Seja ϕ uma aproximação para a função incógnita $\Phi(\mathbf{x})$, dada por:

$$\Phi \approx \phi\,(\mathbf{x}) = \sum_i N_i(\mathbf{x})\phi_i \tag{4}$$

na qual$\{N_i\}$ é uma base funcional, i.é. um conjunto de funções linearmente independentes denominadas de funções de base, as quais podem ser calculadas usando-se, por exemplo, o MEF e o EFGM, dentre muitos outros.

Aplicando-se o Método dos Resíduos Ponderados (MRP) ao problema de valor contorno dado pelas equações (1) a (3) obtemos as seguintes funções resíduo:

$$R_1(\mathbf{x}) = \nabla^2\phi\,(\mathbf{x}) - \rho\,(\mathbf{x}); \qquad R_2(\mathbf{x}) = \phi - \phi_{Dir}; \qquad R_3(\mathbf{x}) = \frac{\partial\phi}{\partial n} - f_{Neu}. \tag{5}$$

Utilizando-se do procedimento de discretização de Galerkin, e aplicando-se a condição de ortogonalidade funcional entre as funções resíduo e as de base, pode-se obter o seguinte funcional I:

$$I = \int_\Omega R_1(\mathbf{x})N_j(\mathbf{x})d\varpi + \int_{\Gamma_{Dir}} R_2(\mathbf{x})N_j(\mathbf{x})d\gamma + \int_{\Gamma_{Neu}} R_3(\mathbf{x})N_j(\mathbf{x})d\gamma = 0 \qquad j = 1..n. \tag{6}$$

Usando-se a identidade de Green e assumindo condições homogêneas de Neumann em $\Gamma_{Neu} \in \Gamma$, para uma escolha adequada de funções de base $N_j(\mathbf{x})$, pode-se chegar à seguinte forma discretizada residual:

$$\int_\Omega \nabla N_j \cdot \nabla\phi\, d\varpi = \int_\Omega N_j\rho\, d\varpi \qquad j = 1..n. \tag{7}$$

Substituindo a Eq. (4) na Eq. (7) e dando continuidade ao procedimento de Galerkin, minimizamos a equação residual resultante com respeito aos coeficientes escalares ϕ_i da aproximação local dada pela Eq. (4), e chegamos ao seguinte sistema linear para as incógnitas ϕ_i:

$$\mathbf{K}\vec{\phi} = \mathbf{b} \tag{8}$$

no qual,

$$[\mathbf{K}]_{ij} = \int_{\Omega} \nabla N_j \cdot \nabla N_i d\varpi, \qquad [\mathbf{b}]_j = \int_{\Omega} N_j \rho_j \, d\varpi, \qquad [\vec{\phi}]_j = \phi_j. \tag{9}$$

A matriz **K** é referida como matriz de rigidez e o vetor **b** como vetor de fontes e consistem nas formas discretas mais recorrentes na Literatura e mais comumente implementadas nos sistemas CAD/CAE para resolução da equação de Poisson.

Para que seja possível a resolução do sistema linear da Eq. (8) é necessário calcular-se as funções de base e suas derivadas espaciais, as quais podem ser obtidas pelo MEF e EFGM, em particular. A resolução deste sistema linear permite obter os valores dos coeficientes nodais ϕ_i, a partir dos quais se pode calcular a aproximação em qualquer ponto arbitrário do domínio usando-se a expansão local dada pela Eq. (4).

2.2 O Método Element Free Galerkin

No EFGM (Belytschko et al., 1994), a aproximação dada pela expansão Eq. (4) é construída com base em um conjunto de N pontos nodais, $\mathbf{x}_i \in \Omega$, provenientes da discretização do domínio Ω. A cada ponto nodal $\mathbf{x}_i$ é associada uma função de ponderação $w(\|\mathbf{x}_i - \mathbf{x}\|) \equiv w_i(\mathbf{x})$, e um parâmetro de aproximação (coeficiente escalar) ϕ_i, que controla a influência dos pontos nodais na aproximação, chamados de parâmetros nodais.

As funções de ponderação podem ser escolhidas arbitrariamente desde que sejam: 1) não-negativas; 2) decresçam monotonicamente à medida que $\|\mathbf{x}_i - \mathbf{x}\|$ cresce; e 3) tenham norma unitária e suporte compacto. Temos empregado com sucesso funções de ponderação singulares construídas a partir da função de Schwarz truncada:

$$w(r_i) = \begin{cases} e^{\frac{r^2}{r^2-1}}, & 0 \le r < 1 \\ 0, & r \ge 1, \end{cases} \qquad \text{emque } r_i = \|\mathbf{x} - \mathbf{x}_i\| / d_{m_i}, \tag{10}$$

e d_{m_i} é o raio de influência. O raio de influência define o alcance da função de ponderação, também chamado de suporte nodal ou domínio de influência nodal, sendo que a superposição dos domínios de influência é o que define a conectividade nodal em cada ponto do domínio.

Diferentes topologias de domínios de influência podem ser definidos, por exemplo, circulares como definido em (10), e retangulares ao definirmos a função de ponderação por meio de um produto tensorial entre as componentes espaciais:

$$w(\|\mathbf{x} - \mathbf{x}_i\|) = w(r_x) \cdot w(r_y), \text{ em que } \qquad r_x = \|x - x_i\| / dm_{x_i} \text{ e } r_y = \|y - y_i\| / dm_{y_i} \tag{11}$$

Considerando que freqüentemente teremos que resolver o problema de inclusão de um ponto de avaliação em domínios de influência, os domínios retangulares são mais comumente empregados por apresentarem menor complexidade computacional.

Podemos escrever uma aproximação local moving least-squares (MLS) para $\Phi(\mathbf{x})$ como:

$$u^h(\mathbf{x}, \bar{\mathbf{x}}) = \sum_{j=1}^{m} p_j(\mathbf{x}) \cdot a_j(\bar{\mathbf{x}}) \equiv \mathbf{p}^T(\mathbf{x}) \cdot \mathbf{a}(\bar{\mathbf{x}}), \tag{12}$$

onde $\mathbf{p}^T(\mathbf{x})$ forma uma base polinomial no espaço das coordenadas com cardinalidade m, e os coeficientes $\mathbf{a}(\mathbf{x})$ podem ser obtidos pela minimização da norma L_2 ponderada:

$$J = \frac{1}{2}\sum_{i=1}^{n} w(\|\mathbf{x}-\mathbf{x}_i\|) \cdot \left[\mathbf{p}^{\mathrm{T}}(\mathbf{x}_i) \cdot \mathbf{a}(\mathbf{x}) - u_i\right]^2 . \tag{13}$$

As funções de ponderação desempenham o papel principal na aproximação EFGM, uma vez que a aproximação construída herdará a maior continuidade dentre os monômios da base polinomial local $\mathbf{p}^{\mathrm{T}}(\mathbf{x})$ e a função de ponderação (Belytschko et al., 1996).

Como a função de Schwarz é de classe de continuidade C^∞, a aproximação obtida será de classe C^∞. Isto significa que a variável de estado (potencial elétrico) e suas derivadas (campo elétrico) serão funções continuamente suaves e deriváveis de classe C^∞, apesar de termos empregado uma base local MLS polinomial linear. Minimizando a norma J na Eq. (13) com respeito aos coeficientes $\mathbf{a}(\mathbf{x})$, e substituindo o resultado na expansão local, Eq. (12), chegamos a:

$$u^h(\mathbf{x}) = \sum_{i=1}^{n} N_i(x) \cdot \phi_i, \tag{14}$$

$$N_i(\mathbf{x}) = \sum_{j}^{m} \mathbf{p}_j(\mathbf{x})\left(\mathbf{A}^{-1}(\mathbf{x})\mathbf{B}(\mathbf{x})\right)_{ji}, \tag{15}$$

sendo as matrizes $\mathbf{A}$(x) e $\mathbf{B}$(x) dadas por $\mathbf{A} = \mathbf{PWP}^T$, $\mathbf{B} = \mathbf{PW}$, e as matrizes $\mathbf{P}$ e $\mathbf{W}$ definidas como usual (Breitkopf et al., 2000).

Além dos valores das funções de forma, a resolução das equações íntegro-diferenciais na Eq. (9) requer o cálculo das derivadas, que podem ser obtidas aplicando-se a regra da cadeia para derivar simbolicamente a Eq. (15) com respeito às variáveis espaciais:

$$\frac{d\mathbf{N}}{dx} = \frac{d\mathbf{p}^T}{dx}\mathbf{A}^{-1}\mathbf{B} + \mathbf{p}^T\frac{d\mathbf{A}^{-1}}{dx}\mathbf{B} + \mathbf{p}^T\mathbf{A}^{-1}\frac{d\mathbf{B}}{dx}. \tag{16}$$

Resta dizer que as funções de forma $N_i(\mathbf{x})$ do EFGM satisfazem o critério de partição de unidade assim como as funções de forma do MEF, contudo, diferentemente do MEF, sem uma escolha adequada das funções de ponderação, as funções de forma do EFGM não satisfazem a propriedade do delta de Kronecker e dão origem a uma aproximação não-interpolante, isto é, os valores das variáveis nodais ϕ_i da Eq. (14) não correspondem aos valores da variável de estado no nó *i*: $u^h(\mathbf{x}_i) \neq \phi_i$ (Belytschko et al., 1994). Com a escolha de uma função peso *singular* pode-se obter funções de forma EFGM que satisfazem a condição do delta de Kronecker, conforme discutido originalmente no trabalho de Breitkopf et al. (2000).

2.3 EFGM e MEF: Algumas analogias conceituais e procedimentais

Quanto às propriedades funcionais da base de funções gerada pelo MEF e pelo EFGM é importante considerar o caráter interpolante inerente a aproximação pelo MEF, o qual pode ser verificado pelo atendimento à propriedade do delta de Kronecker pelas funções de base do MEF. Como dito anteriormente, aproximações interpolantes podem ser obtidas com o EFGM desde que sejam usadas funções peso singulares, as quais darão origem a funções de forma EFGM que atenderão a propriedade do delta de Kronecker, conforme detalhado originalmente no trabalho de Bretkopf et al. (2000) no contexto do EFGM.

Quando a aproximação do EFGM não é interpolante, o uso de técnicas adicionais para imposição das condições de contorno essenciais tais como Multiplicadores de Lagrange e Penalidades é mandatório, além de que o tratamento de interfaces materiais também requer técnicas específicas (Cordes e Moran, 1996; Marques et al., 2007). Diferentemente, no MEF a imposição das condições de contorno de Dirichlet no sistema linear da Eq. (8) é eficientemente feita pela substituição direta dos valores impostos, e o tratamento de interfaces materiais acontece naturalmente, à medida que a malha respeita os limites do modelo geométrico e físico.

Nestas condições, o uso da abordagem interpolante do EFGM combinada com a técnica de truncamento de domínios modificada (Marques et al., 2007) é bastante atrativo tendo em vista as seguintes características: 1) o atendimento ao delta de Kronecker pelas funções de forma habilita o uso da técnica de substituição direta para imposição das condições de Dirichlet, sendo suficiente um bom refinamento nos contornos para eficácia da técnica; 2) um bom refinamento das interfaces materiais associado ao uso da simples técnica de truncamento de domínios modificada tem fornecido resultados excelentes comparados a resultados do MEF.

A principal analogia conceitual a nosso ver, é relativa ao "elemento finito" enquanto entidade conceitual inerente a formulação do MEF e o "elemento difuso" enquanto entidade conceitual inerente ao EFGM, muito embora este termo não tenha sido muito referido em grande parte dos trabalhos com o EFGM. O elemento difuso (Nayroles et al., 1992) consiste no conjunto de nós que participam da aproximação em um dado ponto de avaliação, e é a partir das funções de forma e suas derivadas associadas a esses nós que a aproximação é realizada, assim como para um ponto de avaliação arbitrário no interior de um elemento finito. Por outro lado, os elementos finitos provêm de uma malha e, portanto atendem a propriedades topológicas como não-intersecção, regularidade e conectividade, compondo uma estrutura geométrica bem definida e fixa ao longo de uma iteração, ao passo que os elementos difusos são entidades conceituais instanciadas para o cálculo das contribuições nodais, são não-persistentes, e caracterizados principalmente por suas propriedades topológicas funcionais, sendo sua descrição geométrica pouco relevante.

Ambos MEF e EFGM necessitam da utilização de um integrador para calcular as contribuições integrais nodais dadas pelas equações Eq. (9). No caso do MEF, os elementos finitos constituem uma verdadeira estrutura de células de integração para aplicação da quadratura de Gauss; no EFGM, uma estrutura de células de integração deve ser criada.

3 MODELAGEM COMPUTACIONAL DO EFGM NO SISTEMA LEVSOFT

3.1 O LEVSOFT

Assim como descrito no site do Instituto de Estudos Avançados do Centro Técnico Aeroespacial (2007), o projeto LEVSOFT iniciou-se em 1998 com o desenvolvimento de um *software* para a análise e projeto de dispositivos óticos e componentes integrados, baseado no Método dos Elementos Finitos em 2D (2 dimensões). O desenvolvimento inicial, financiado pela FAPESP (Dezembro de 1998 a Janeiro de 2003) permitiu a implementação dos módulos pré-processador, processador e pós-processador do sistema, que possibilitam o processo de desenho da geometria, cálculo e análise dos campos eletromagnéticos calculados.

O código-fonte do LEVSOFT está implementado em linguagem C^{++}, utilizando-se do paradigma orientado a objetos possibilitado pela mesma, e possui alguns módulos com funcionalidades adicionais implementados em Matlab. Como principal proposta, o LEVSOFT foi desenvolvido como uma plataforma para a inclusão de outros fenômenos físicos conduzindo a um aumento progressivo da natureza dos estudos compreendidos pelo *software*.

Hoje em dia, o LEVSOFT permite o projeto de dispositivos eletromagnéticos de alta freqüência tais como a simulação de fibras óticas, componentes óticos integrados baseados em efeitos acoplados incluindo eletro-ótica, magneto-ótica, termo-ótica e dispositivos microondas, também permite a análise de dispositivos eletromagnéticos de baixa freqüência compreendendo fenômenos eletrostáticos, magnetostáticos e eletrocinéticos, bem como o estudo de problemas térmicos em regime estacionário e transiente. Além disso, existe uma implementação do LEVSOFT para suportar o EFGM codificada parcialmente no paradigma estruturado desenvolvida para rápida prototipagem e testes, a qual foi tomada como *toy-problem* no desenvolvimento desse trabalho.

3.2 Análise do Protótipo LEVSOFT/EFGM e Modelagem da Nova Classe

3.2.1 A Arquitetura das Principais Classes do EFGM e MEF no LEVSOFT

O diagrama de classes representado na Figura 1 detalha a estrutura das principais classes responsáveis pelos cálculos realizados com o MEF e o EFGM para resolução da equação de Poisson e com o Particle In Cell (PIC) para simulação de plasmas implementadas no módulo de cálculo do LEVSOFT, as classes CLevFEM_Poisson, CLevEFGM_Poison, e CLevPIC.

A classe FiniteElement modela o elemento finito enquanto elemento de malha: seus vértices, seus elementos vizinhos, as propriedades físicas e a presença de fontes, e fornecem, a partir de seus métodos, informações para busca geométrica em malha, cálculo das dimensões geométricas do elemento, para cálculo do centro geométrico do elemento além de funcionalidades básicas de acesso. As classes SimplexNodalElement 1D, ____2D e ____3D são derivadas da classe NodalElement que por sua vez deriva de FiniteElement, de modo que as informações do modelo geométrico e físico são acessíveis pelas classes SimplexNodalElement__, e ao mesmo tempo hierarquicamente protegidos.

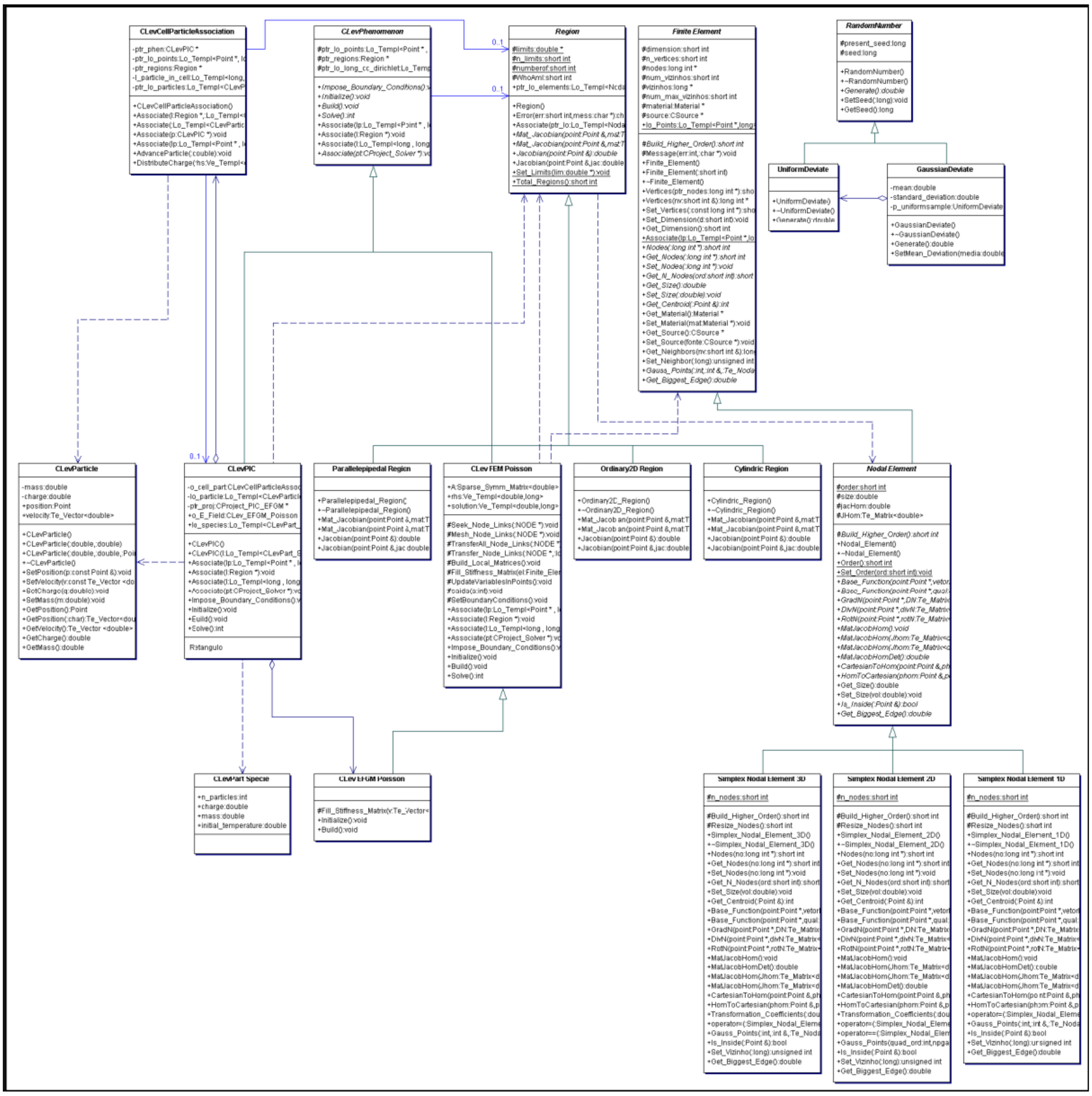

Figura 1. Diagrama de Classes do núcleo de cálculo do MEF e PIC no sistema LEVSOFT.

Analisando a construção das matrizes locais e global do MEF (CLevFEM_Poisson::Build( )), observa-se que a classe que dá suporte ao cálculo das funções de forma e do processo de quadratura de Gauss é uma classe derivada da classe Nodal Element, SimplexNodalElement2D. Esta classe se beneficia da herança das funcionalidades de acesso e recuperação de informação da classe FiniteElement (a malha) e fornece o cálculo dos valores da função de forma associada a um dado nó nas posições dos pontos de Gauss dentro do suporte nodal. O elemento difuso no EFGM desempenha o mesmo papel, e usando-se a estrutura de malha para buscar e construir a relação de conectividade sobre os pontos de avaliação a partir dos elementos de malha em que se encontram, obtém-se uma implementação bastante eficiente para esse procedimento.

Analisando-se mais cuidadosamente a hierarquia dessas classes do sistema LEVSOFT, pode-se notar que a classe Finite Element está mais relacionada a funcionalidades inerentes à malha, ao passo que a classe que dá suporte à montagem das matrizes locais e globais do MEF é a classe derivada Nodal Element, mais precisamente suas especializações SimplexNodalElement. A proposta de uso da classe Finite Element para tornar mais eficiente a realização de tarefas específicas do EFGM (Marques et al. 2007; Seidl e Schmidt, 2007) parece, de fato, ter lugar na arquitetura de software do módulo de cálculo em estudo, capaz de comportar de forma coerente o aproveitamento da malha na implementação do EFGM orientado a objetos no sistema LEVSOFT, tanto do ponto de vista conceitual Matemático quanto com respeito ao paradigma OO.

A prototipagem rápida do código estruturado EFGM integrado ao LEVSOFT foi realizada com a derivação da classe CLevFEM_Poisson, que deu origem à classe CLevEFGM_Poisson na qual os métodos Initialize() e Build() foram sobrecarregados em suas funcionalidades já que, além de suas atribuições usuais como as associações e inicializações ficaram também encarregados das trocas de dados entre o código estruturado do EFGM e o código OO do LEVSOFT.

3.2.2 Tratamento dos Métodos Initialize() e Build() de CLevEFGM_Poisson

Para rápida prototipagem os métodos Initialize() e Build() da classe CLevEFGM_Poisson, herdados da classe CLevFEM_Poisson, foram especializados e, de certa forma, sobrecarregados para lidar com a incorporação do código estruturado do EFGM. A Por outro lado, outros métodos ligados à imposição das condições de contorno e à resolução do sistema linear não necessitaram de nenhuma re-implementação.

Resumidamente, o método Initialize( ) criava e inicializava as estruturas de dados usadas pelo código estruturado do EFGM a partir das informações do modelo geométrico contidas nas estruturas de dados O.O. do LEVSOFT. Neste caso, as chamadas às estruturas de dados do código estruturado devem ser cuidadosamente substituídas pela chamada a métodos de classes pertinentemente relacionadas como as relativas aos nós e à malha de elementos finitos.

O método Build( ) utilizava de várias chamadas à funções estruturadas para recuperar as informações do elemento difuso para montagem das matrizes de contribuição local e global, bem como conversões de tipos de dados entre o código estruturado e o OO. O cálculo das contribuições locais e global pelo MEF é realizada pela classe SimplexNodalElement e ao considerar-se a implementação do método Build( ) na classe CLevFEM_Poisson, além das relações conceituais aqui discutidas, percebe-se que é quase natural a concepção de uma nova classe denominada SimplexNodalDifuseElement_2D, ao mesmo nível hierárquico que as classes SimplexNodalElement_1D, __2D e __3D. A maior parte desses cálculos em Build( ) da classe CLevEFGM_Poisson passa a ser realizado de forma muito semelhante ao método Build( ) da classe CLevEFGM_Poisson quando criamos a classe SimplexNodalDifuseElement_2D, visto que provê, correspondentemente, as mesmas funcionalidades (métodos) da classe SimplexNodalElement_#D, com mínimas alterações, resultando numa melhor uniformidade, coerência e clareza no código resultante.

É importante dizer que o uso das informações e funcionalidades da malha de elementos finitos não impõe limitações à aplicação do EFGM quanto à qualidade da malha, como ocorrem no MEF, tendo em vista que, diferentemente, no EFGM a conectividade nodal que determina as contribuições locais é obtida pelo critério de inclusão quanto ao domínio de influência nodal e não a partir da conectividade da malha, de modo que a presença de elementos distorcidos irá influir apenas no processo de integração e não no cálculo dos valores das funções de forma e suas derivadas. Além disso, conforme verificado por Seidl e Schmidt (2006), para uma mesma malha relativamente degenerada, o impacto dos elementos distorcidos nas aproximações pelo EFGM é muito menor que nas aproximações pelo MEF.

4 CONCLUSÕES

Este trabalho detalha um processo de engenharia de software em curso voltado a implementação de uma abordagem interpolante do EFGM ao sistema CAD/CAE denominado LEVSOFT que implementa originalmente o MEF segundo o paradigma OO. O protótipo rápido inicial do sistema LEVSOFT integrado ao código estruturado do EFGM foi descrito e permitiu destacar a hierarquia de classes do LEVSFOT bem como discutirmos a distribuição das funcionalidades entre as classes que implementam os métodos. Evidenciadas algumas analogias teóricas conceituais e procedimentais entre os métodos em questão, especialmente os conceitos de elemento difuso e elemento finito, fundamentamos a proposta de uma nova classe denominada SimplexNodalDifuseElement_2D situada ao mesmo nível das classes SimplexNodalElement_1D, __2D e __3D e com semelhanças conceituais claras, o que permitiu melhor coerência conceitual entre as classes que implementam os dois métodos e maior uniformidade entre os procedimentos empregados em cada método. O caráter interpolante da aproximação foi considerado e os métodos de acesso aos elementos de malha foram explorados para maior eficiência da tarefa de determinação da conectividade nos pontos de integração de Gauss, bem como os elementos da malha foram utilizados como células de quadratura para integração das equações íntegro-diferenciais.

AGRADECIMENTOS

Agrademos à Fundação de Amparo a Pesquisa do Estado de Mato Grosso (FAPEMAT, nº 002.229/2007) pelo apoio financeiro e ao grupo do Laboratório de Engenharia Virtual – LEV do IEAv/CTA, em especial ao Dr. Angelo Passaro e a Dra. Nancy Mieko Abe pelas contribuições com o sistema LEVSOFT.

REFERÊNCIAS

Belytschko, T., Lu, Y. Y. e Gu, L., Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering,* 37:229-256, 1994.

Belytschko, T., Krongauz, Y., Organ, D., Fleming, M. e Krysl, P. Meshless Methods: An overview and recent developments, *Computer Methods in Applied Mechanics and Engineering,* 139:3-47, 1996.

Belytschko, T. e Krysl, P., ESFLIB: A library to compute the element free Galerkin shape functions. *Comput. Meth. Appl. Mech. Engng*., 190:2181-205, 2001.

Breitkopf, P., Rassineux, A., Touzot, G. e Villon, P., Explicit form and efficient computation of MLS shape functions and their derivatives, *Int. Journal for Num. Methods in Engng.,* 48:451-466, 2000.

Cordes, L. W. e Moran, B. Treatment of material discontinuity in the Element-Free Galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 139:75-89, 1996.

Frauenfelder, P. e Lage C., Concepts-An Object-Oriented Software Package for Partial Differential Equations. *Mathematical Modelling and Num. Analysis, 36(5):937–951, 2002. DOI: 10.1051/m2an:2002036.*

Marques, G.N., Machado, J.M., Verardi, S.L.L., Stephany, S. e Preto, A.J. Interpolating EFGM for computing continuous and discontinuous electromagnetic fields. *COMPEL – The International Journal for Computation and Mathematics in Electrical and Electronic Engineering,* 26(5):1411-1438, 2007.

Nayroles, B. Touzot G. e Villon, P. Generalizing the finite element method: diffuse approximation and diffuse elements. *Computational Mechanics,* 10:307-318, 1992.

Paes, A.C.J., Abe, N.M., Serrão, V.A. e Passaro, A., Análise de um Mecanismo de Aceleração de Feixe Utilizando um Modelo de Partículas Autoconsistente e o Método dos Elementos Finitos. In**: V** COMPUMAG: Congresso Brasileiro de Eletromagnetismo, 2002, Gramado. **Anais...** Gramado: [s.n], 2002.

Seidl, A. e Schmidt, Th., Implementation of Meshless FEM for Engineering Applications*, Proceedings of World Academy of Science, Engineering and Technology, 23:1-5, August 2007, ISSN 1307-6884.*

Seidl, A. e Schmidt, Th., Object oriented approach to consistent implementation of Meshless and classical FEM, *Systemics, Cybernetics and Informatics*, 4(1):58-64, 2006.